

Meeting

version 1.1

BTS SNIR LaSalle Avignon 2021

Table des matières

1	Application Android	2
1.1	Table des matières	2
1.2	Informations	2
2	Planification	2
3	Protocole Meeting 2021	3
4	README	4
4.1	Meeting	4
4.1.1	Présentation	4
4.1.2	Informations	5
5	A propos	5
6	Licence GPL	5
7	Documentation des espaces de nommage	5
7.1	Paquetage com	5
7.2	Paquetage com.lasalle	5
7.3	Paquetage com.lasalle.meeting	6
8	Documentation des classes	6
8.1	Référence de la classe com.lasalle.meeting.AffichageEspaceDeTravail	6
8.1.1	Description détaillée	8
8.1.2	Documentation des fonctions membres	8
8.1.3	Documentation des données membres	16
8.2	Référence de la classe com.lasalle.meeting.Communication	17
8.2.1	Description détaillée	19
8.2.2	Documentation des constructeurs et destructeur	19
8.2.3	Documentation des fonctions membres	20
8.2.4	Documentation des données membres	26
8.3	Référence de la classe com.lasalle.meeting.EspaceDeTravail	32
8.3.1	Description détaillée	34

8.3.2	Documentation des constructeurs et destructeur	34
8.3.3	Documentation des fonctions membres	34
8.3.4	Documentation des données membres	44
8.4	Référence de la classe <code>com.lasalle.meeting.EspaceDeTravailAdaptateur</code>	48
8.4.1	Description détaillée	49
8.4.2	Documentation des constructeurs et destructeur	49
8.4.3	Documentation des fonctions membres	49
8.4.4	Documentation des données membres	51
8.5	Référence de la classe <code>com.lasalle.meeting.IHMMeeting</code>	51
8.5.1	Description détaillée	54
8.5.2	Documentation des fonctions membres	54
8.5.3	Documentation des données membres	72
8.6	Référence de la classe <code>com.lasalle.meeting.ModificationEspaceDeTravail</code>	77
8.6.1	Description détaillée	78
8.6.2	Documentation des fonctions membres	78
8.6.3	Documentation des données membres	81
8.7	Référence de la classe <code>Runnable</code>	82
8.8	Référence de la classe <code>com.lasalle.meeting.EspaceDeTravailAdaptateur.ViewHolder</code>	82
8.8.1	Description détaillée	82
9	Documentation des fichiers	83
9.1	Référence du fichier <code>AffichageEspaceDeTravail.java</code>	83
9.1.1	Description détaillée	83
9.2	<code>AffichageEspaceDeTravail.java</code>	83
9.3	Référence du fichier <code>Communication.java</code>	88
9.3.1	Description détaillée	88
9.4	<code>Communication.java</code>	88
9.5	Référence du fichier <code>EspaceDeTravail.java</code>	92
9.5.1	Description détaillée	92
9.6	<code>EspaceDeTravail.java</code>	93
9.7	Référence du fichier <code>EspaceDeTravailAdaptateur.java</code>	96
9.7.1	Description détaillée	96
9.8	<code>EspaceDeTravailAdaptateur.java</code>	97
9.9	Référence du fichier <code>IHMMeeting.java</code>	98
9.9.1	Description détaillée	98
9.10	<code>IHMMeeting.java</code>	99
9.11	Référence du fichier <code>ModificationEspaceDeTravail.java</code>	108
9.11.1	Description détaillée	108
9.12	<code>ModificationEspaceDeTravail.java</code>	108
9.13	Référence du fichier <code>Planification.md</code>	110
9.14	<code>Planification.md</code>	110
9.15	Référence du fichier <code>Protocole.md</code>	110
9.16	<code>Protocole.md</code>	110
9.17	Référence du fichier <code>README.md</code>	112
9.18	<code>README.md</code>	112

1 Application Android

1.1 Table des matières

- [README](#)
- [A propos](#)
- [Licence GPL](#)

1.2 Informations

Auteur

KELLER-LAVALLEE Joachim <joachim.kellerlavallee@gmail.com>

Date

2021

Version

1.1

Voir également

<https://svn.riouxsvn.com/meeting-2021/>

2 Planification

Fonctionnalité	Priorité	Itération
Visualiser la liste des espaces de travail	Haute	1
Visualiser un espace de travail (nom, lieu, description, superficie, disponibilité, durée d'occupation, température et indice de confort)	Haute	1
Communiquer avec les portiers	Haute	2
Réserver un espace de travail avec durée d'occupation	Moyenne	2
Augmenter la durée d'occupation	Moyenne	2
Libérer un espace de travail	Haute	2
Rechercher un espace de travail par nom, disponibilité et/ou indice de confort	Moyenne	3
Editer un espace de travail	Haute	3
Visualiser les favoris	Basse	3
Ajouter/retirer un espace de travail aux favoris	Basse	3

Itération 1

- Visualiser la liste des espaces de travail
- Visualiser un espace de travail (nom, lieu, description, superficie, disponibilité, température et indice de confort)

Itération 2

- Communiquer avec les portiers
- Réserver un espace de travail avec durée d'occupation
- Augmenter la durée d'occupation
- Libérer un espace de travail

Itération 3

- Rechercher un espace de travail par nom, disponibilité et/ou indice de confort
- Editer un espace de travail
- Visualiser les favoris
- Ajouter/retirer un espace de travail aux favoris

3 Protocole Meeting 2021

Description générale

Protocole orienté ASCII

Délimiteurs :

- début : \$
- champs : ;
- fin : \r\n

Adresse de multicast des portiers : 239.0.0.x

Format des trames de requêtes Application -> Portier(s) en multicast : \$GET;idRequete\r\n

Format des trames d'actualisations Application -> Portier(s) en unicast : \$SET;idRequete\r\n

idRequete	Signification	Requêtes	Actualisations
1	informations	X	X
2	état	X	
3	disponibilité	X	X

Les différents champs d'une trame de réponse :

champ	description
nomSalle	string (le nom de la salle)
description	string (description de la salle)
emplacement	string (l'emplacement de la salle)
disponibilité	0 = occupé et 1 = libre
niveauDeConfort	de -3 à 3 (voir cahier des charges)
température	en degré
surface	en m²

Demande les informations des portiers

Application -> Portier(s) en multicast : \$GET;1\r\n

Portier -> Application : \$nomSalle;description;emplacement;surface;disponibilité;niveauDeConfort;température\r\n

— Nombre de champs : 7

Demander l'état des portiers

Application -> Portier(s) en multicast : \$GET;2\r\n

Portier -> Application : \$nomSalle;disponibilité;niveauDeConfort;température\r\n

— Nombre de champs : 4

Demander la disponibilité des portiers

Application -> Portier(s) en multicast : \$GET;3\r\n

Portier -> Application : \$nomSalle;disponibilité\r\n

— Nombre de champs : 2

Actualiser les informations d'un portier

Application -> Portier en unicast : \$SET;1;nomSalle;description;emplacement;surface\r\n

Portier -> Application : \$nomSalle;ok\r\n

— Nombre de champs : 2

Actualiser la disponibilité d'un portier

— Pour réserver :

Application -> Portier en unicast : \$SET;3;0\r\n

Retour :

Portier -> Application : \$nomSalle;code;OK\r\n

Nombre de champs : 4

Portier -> Application : \$nomSalle;;ERREUR\r\n

Nombre de champs : 3

— Pour libérer :

Application -> Portier en unicast : \$SET;3;1;code\r\n

Portier -> Application : \$nomSalle;;OK\r\n

Nombre de champs : 3

Portier -> Application : \$nomSalle;;ERREUR\r\n

Nombre de champs : 3

4 README

4.1 Meeting

4.1.1 Présentation

Meeting est une application mobile qui communique avec un portier connecté via une liaison wifi et qui permet de :

- Visualiser la liste des espaces de travail
- Visualiser les données de l'espace de travail
- Réserver et libérer un espace de travail
- Editer un espace de travail
- Rechercher un espace de travail par mot-clé, disponibilité et niveau de confort
- Gérer les favoris

4.1.2 Informations

Auteur

KELLER-LAVALLEE Joachim <joachim.kellerlavallee@gmail.com>

Date

2021

Version

1.1

Voir également

<https://svn.riouxsvn.com/meeting-2021/>

5 A propos

Auteur

KELLER-LAVALLEE Joachim <joachim.kellerlavallee@gmail.com>

6 Licence GPL

This program is free software ; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation ; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY ; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program ; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

7 Documentation des espaces de nommage

7.1 Paquetage com

Paquetages

— package [lasalle](#)

7.2 Paquetage com.lasalle

Paquetages

— package [meeting](#)

7.3 Paquetage com.lasalle.meeting

Classes

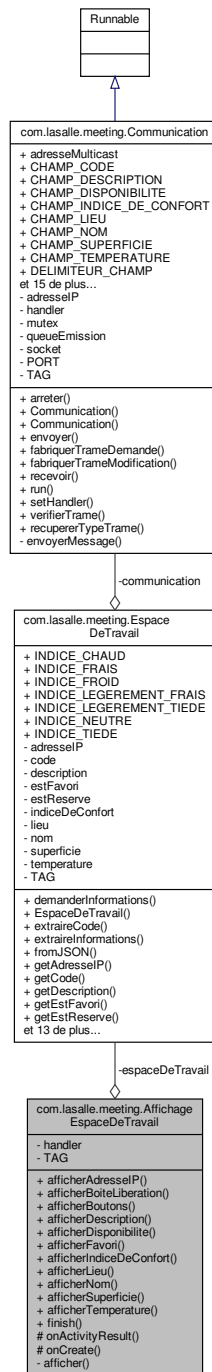
- class [AffichageEspaceDeTravail](#)
L'activité d'affichage d'un espace de travail de l'application Meeting.
- class [Communication](#)
[Communication](#) entre l'application et le portier.
- class [EspaceDeTravail](#)
L'espace de travail.
- class [EspaceDeTravailAdaptateur](#)
L'affichage d'un espace de travail dans la liste des espaces de travail sur la page d'accueil.
- class [IHMMeeting](#)
L'activité principale de l'application Meeting.
- class [ModificationEspaceDeTravail](#)
L'activité de modification d'un espace de travail de l'application Meeting.

8 Documentation des classes

8.1 Référence de la classe com.lasalle.meeting.AffichageEspaceDeTravail

L'activité d'affichage d'un espace de travail de l'application Meeting.

Grphe de collaboration de com.lasalle.meeting.AffichageEspaceDeTravail :



Fonctions membres publiques

- void `afficherAdresseIP ()`
Affiche l'adresse IP du portier.
- void `afficherBoiteLiberation ()`
Affiche la boîte de dialogue de saisie du code de libération.
- void `afficherBoutons ()`
Affiche les boutons "Réserver", "Libérer", "Editer les informations", "Ajouter aux favoris" et "Retirer des favoris".
- void `afficherDescription ()`
Affiche la description de l'espace de travail.
- void `afficherDisponibilite ()`

- void [afficherFavori](#) ()
Affiche la disponibilité de l'espace de travail.
- void [afficherIndiceDeConfort](#) ()
Affiche si l'espace de travail est en favori.
- void [afficherLieu](#) ()
Affiche l'indice de confort de l'espace de travail.
- void [afficherNom](#) ()
Affiche le lieu de l'espace de travail.
- void [afficherSuperficie](#) ()
Affiche le nom de l'espace de travail.
- void [afficherTemperature](#) ()
Affiche la superficie de l'espace de travail.
- void [finish](#) ()
Affiche la température de l'espace de travail.
- void [finish](#) ()
Termine l'activité d'affichage d'un espace de travail.

Fonctions membres protégées

- void [onActivityResult](#) (int requestCode, int resultCode, Intent data)
Traite le retour de l'activité de modification d'un espace de travail.
- void [onCreate](#) (Bundle savedInstanceState)
Méthode appelée à la création de l'activité

Fonctions membres privées

- void [afficher](#) ()
Affiche les propriétés de l'espace de travail.

Attributs privés

- [EspaceDeTravail](#) [espaceDeTravail](#)
L'espace de travail.
- Handler [handler](#)

Attributs privés statiques

- static final String [TAG](#) = "_AffichageEspaceTravail"
TAG pour les logs.

8.1.1 Description détaillée

L'activité d'affichage d'un espace de travail de l'application Meeting.

Définition à la ligne 34 du fichier [AffichageEspaceDeTravail.java](#).

8.1.2 Documentation des fonctions membres

8.1.2.1 `afficher()`

```
void com.lasalle.meeting.AffichageEspaceDeTravail.afficher ( ) [private]
```

Affiche les propriétés de l'espace de travail.

Définition à la ligne 64 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.AffichageEspaceDeTravail.afficherAdresseIP\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherBoutons\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherDescription\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherDisponibilite\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherFavori\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherIndiceDeConfort\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherLieu\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherNom\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherSuperficie\(\)](#), et [com.lasalle.meeting.AffichageEspaceDeTravail.afficherTemperature\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.onCreate\(\)](#).

```
00065     {
00066         afficherAdresseIP ();
00067         afficherNom ();
00068         afficherLieu ();
00069         afficherDescription ();
00070         afficherSuperficie ();
00071         afficherTemperature ();
00072         afficherIndiceDeConfort ();
00073         afficherDisponibilite ();
00074         afficherFavori ();
00075         afficherBoutons ();
00076     }
```

8.1.2.2 `afficherAdresseIP()`

```
void com.lasalle.meeting.AffichageEspaceDeTravail.afficherAdresseIP ( )
```

Affiche l'adresse IP du portier.

Définition à la ligne 81 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.getAdresseIP\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficher\(\)](#).

```
00082     {
00083         TextView affichageAdresseIP = (TextView)findViewById(R.id.affichageAdresseIP);
00084         affichageAdresseIP.setText(espaceDeTravail.getAdresseIP());
00085
00086         Log.d(TAG, "afficherAdresseIP() " + espaceDeTravail.
00087             getAdresseIP());
00087     }
```

8.1.2.3 afficherBoiteLiberation()

```
void com.lasalle.meeting.AffichageEspaceDeTravail.afficherBoiteLiberation ( )
```

Affiche la boîte de dialogue de saisie du code de libération.

Définition à la ligne 327 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.liberer\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherBoutons\(\)](#).

```
00328     {
00329         AlertDialog.Builder boiteLiberation = new AlertDialog.Builder(this);
00330         boiteLiberation.setTitle("Libérer l'espace de travail");
00331         boiteLiberation.setMessage("Saisissez le code pour libérer l'espace de travail :");
00332         LayoutInflater inflater = this.getLayoutInflater();
00333         View vue = inflater.inflate(R.layout.boite_liberation, null);
00334         boiteLiberation.setView(vue);
00335
00336         boiteLiberation.setPositiveButton("Libérer", new DialogInterface.OnClickListener()
00337         {
00338             public void onClick(DialogInterface dialog, int which)
00339             {
00340                 EditText saisieCode = (EditText) ((AlertDialog) dialog).findViewById(R.id.saisieCode);
00341                 Log.d(TAG, "onClick() code = " + saisieCode.getText().toString());
00342                 espaceDeTravail.liberer(saisieCode.getText().toString());
00343             }
00344         });
00345         boiteLiberation.setNegativeButton("Annuler", new DialogInterface.OnClickListener()
00346         {
00347             public void onClick(DialogInterface dialog, int which)
00348             {
00349             }
00350         });
00351     });
00352     boiteLiberation.show();
00353 }
00354 }
```

8.1.2.4 afficherBoutons()

```
void com.lasalle.meeting.AffichageEspaceDeTravail.afficherBoutons ( )
```

Affiche les boutons "Réserver", "Libérer", "Editer les informations", "Ajouter aux favoris" et "Retirer des favoris".

Définition à la ligne 232 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.AffichageEspaceDeTravail.afficherBoiteLiberation\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherDisponibilite\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherFavori\(\)](#), [com.lasalle.meeting.EspaceDeTravail.getEstFavori\(\)](#), [com.lasalle.meeting.EspaceDeTravail.getEstReserve\(\)](#), [com.lasalle.meeting.EspaceDeTravail.initialiserCommunication\(\)](#), [com.lasalle.meeting.EspaceDeTravail.reserver\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.setEstFavori\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficher\(\)](#).

```
00233     {
00234         Button boutonReserver = (Button) findViewById(R.id.boutonReserver);
00235         Button boutonLiberer = (Button) findViewById(R.id.boutonLiberer);
00236         Button boutonModifier = (Button) findViewById(R.id.boutonModifier);
00237         Button boutonAjouterFavori = (Button) findViewById(R.id.boutonAjouterFavori);
00238         Button boutonRetirerFavori = (Button) findViewById(R.id.boutonRetirerFavori);
00239
00240         boutonReserver.setOnClickListener(
00241             new View.OnClickListener()
00242             {
00243                 public void onClick(View v)
00244                 {
00245                     espaceDeTravail.reserver();
00246                     afficherDisponibilite();
00247                     afficherBoutons();
00248                 }
00249             }
00250         );
00251         boutonLiberer.setOnClickListener(
00252             new View.OnClickListener()
00253             {
00254                 public void onClick(View v)
00255                 {
00256                     afficherBoiteLiberation();
00257                 }
00258             }
00259         );
00260         boutonModifier.setOnClickListener(
00261             new View.OnClickListener()
00262             {
00263                 public void onClick(View v)
00264                 {
00265                     afficherFavori();
00266                 }
00267             }
00268         );
00269         boutonAjouterFavori.setOnClickListener(
00270             new View.OnClickListener()
00271             {
00272                 public void onClick(View v)
00273                 {
00274                     ajouterFavori();
00275                 }
00276             }
00277         );
00278         boutonRetirerFavori.setOnClickListener(
00279             new View.OnClickListener()
00280             {
00281                 public void onClick(View v)
00282                 {
00283                     retirerFavori();
00284                 }
00285             }
00286         );
00287     }
```

```

00248         }
00249     }
00250 };
00251
00252 boutonLiberer.setOnClickListener(
00253     new View.OnClickListener()
00254     {
00255         public void onClick(View v)
00256         {
00257             afficherBoiteLiberation();
00258             afficherDisponibilite();
00259             afficherBoutons();
00260         }
00261     }
00262 );
00263
00264 if(!espaceDeTravail.getEstReserve())
00265 {
00266     boutonReserver.setVisibility(View.VISIBLE);
00267     boutonLiberer.setVisibility(View.GONE);
00268 }
00269 else
00270 {
00271     boutonReserver.setVisibility(View.GONE);
00272     boutonLiberer.setVisibility(View.VISIBLE);
00273 }
00274
00275 boutonModifier.setOnClickListener(
00276     new View.OnClickListener()
00277     {
00278         public void onClick(View v)
00279         {
00280             Intent intent = new Intent(AffichageEspaceDeTravail.this, ModificationEspaceDeTravail.
class);
00281             espaceDeTravail.initialiserCommunication(null);
00282             intent.putExtra("unEspaceDeTravail", (Serializable)
espaceDeTravail);
00283             startActivityForResult(intent, 0);
00284         }
00285     }
00286 );
00287
00288 boutonAjouterFavori.setOnClickListener(
00289     new View.OnClickListener()
00290     {
00291         public void onClick(View v)
00292         {
00293             espaceDeTravail.setEstFavori(true);
00294             afficherFavori();
00295             afficherBoutons();
00296         }
00297     }
00298 );
00299
00300 boutonRetirerFavori.setOnClickListener(
00301     new View.OnClickListener()
00302     {
00303         public void onClick(View v)
00304         {
00305             espaceDeTravail.setEstFavori(false);
00306             afficherFavori();
00307             afficherBoutons();
00308         }
00309     }
00310 );
00311
00312 if(!espaceDeTravail.getEstFavori())
00313 {
00314     boutonAjouterFavori.setVisibility(View.VISIBLE);
00315     boutonRetirerFavori.setVisibility(View.GONE);
00316 }
00317 else
00318 {
00319     boutonAjouterFavori.setVisibility(View.GONE);
00320     boutonRetirerFavori.setVisibility(View.VISIBLE);
00321 }
00322 }

```

8.1.2.5 afficherDescription()

```
void com.lasalle.meeting.AffichageEspaceDeTravail.afficherDescription ( )
```

Affiche la description de l'espace de travail.

Définition à la ligne 114 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.getDescription\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficher\(\)](#).

```
00115     {
00116         TextView affichageDescription = (TextView)findViewById(R.id.affichageDescription);
00117         affichageDescription.setText(espaceDeTravail.
affichageDescription.setText(espaceDeTravail.
getDescription());
00118
00119         Log.d(TAG, "afficherDescription() " + espaceDeTravail.
affichageDescription.setText(espaceDeTravail.
getDescription());
00120     }
```

8.1.2.6 afficherDisponibilite()

```
void com.lasalle.meeting.AffichageEspaceDeTravail.afficherDisponibilite ( )
```

Affiche la disponibilité de l'espace de travail.

Définition à la ligne 209 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.getEstReserve\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficher\(\)](#), et [com.lasalle.meeting.AffichageEspaceDeTravail.afficher↵ Boutons\(\)](#).

```
00210     {
00211         TextView affichageDisponibilite = (TextView)findViewById(R.id.affichageDisponibilite);
00212
00213         if(!espaceDeTravail.getEstReserve())
00214         {
00215             Log.d(TAG, "afficherDisponibilite() Libre");
00216             affichageDisponibilite.setText("Libre");
00217             affichageDisponibilite.setTextColor(Color.parseColor("#00FF00")); // Color.rgb(0,255,0)
00218         }
00219         else
00220         {
00221             Log.d(TAG, "afficherDisponibilite() Occupé");
00222             affichageDisponibilite.setText("Occupé");
00223             affichageDisponibilite.setTextColor(Color.rgb(255,0,0));
00224         }
00225
00226         Log.d(TAG, "afficherDisponibilite() " + espaceDeTravail.
affichageDisponibilite.setText(espaceDeTravail.
getEstReserve());
00227     }
```

8.1.2.7 afficherFavori()

```
void com.lasalle.meeting.AffichageEspaceDeTravail.afficherFavori ( )
```

Affiche si l'espace de travail est en favori.

Définition à la ligne 188 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.getEstFavori\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficher\(\)](#), et [com.lasalle.meeting.AffichageEspaceDeTravail.afficher↵ Boutons\(\)](#).

```
00189     {
00190         ImageView iconeFavori = (ImageView)findViewById(R.id.iconeFavori);
00191
00192         if(!espaceDeTravail.getEstFavori())
00193         {
00194             Log.d(TAG, "afficherFavori() Non favori");
00195             iconeFavori.setVisibility(View.INVISIBLE);
00196         }
00197         else
00198         {
00199             Log.d(TAG, "afficherFavori() Favori");
00200             iconeFavori.setVisibility(View.VISIBLE);
00201         }
00202
00203         Log.d(TAG, "afficherFavori() " + espaceDeTravail.
affichageFavori.setText(espaceDeTravail.
getEstFavori());
00204     }
```

8.1.2.8 afficherIndiceDeConfort()

```
void com.lasalle.meeting.AffichageEspaceDeTravail.afficherIndiceDeConfort ( )
```

Affiche l'indice de confort de l'espace de travail.

Définition à la ligne 147 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.getIndiceDeConfort\(\)](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_CHAUD](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_FRAIS](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_FROID](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_LEGEREMENT_FRAIS](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_LEGEREMENT_TIEDE](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_NEUTRE](#), et [com.lasalle.meeting.EspaceDeTravail.INDICE_TIEDE](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficher\(\)](#).

```
00148     {
00149         TextView affichageIndiceDeConfort = (TextView)findViewById(R.id.affichageIndiceDeConfort);
00150
00151         switch(espaceDeTravail.getIndiceDeConfort())
00152         {
00153             case EspaceDeTravail.INDICE_CHAUD:
00154                 affichageIndiceDeConfort.setText("Chaud");
00155                 break;
00156
00157             case EspaceDeTravail.INDICE_TIEDE:
00158                 affichageIndiceDeConfort.setText("Tiède");
00159                 break;
00160
00161             case EspaceDeTravail.INDICE_LEGEREMENT_TIEDE:
00162                 affichageIndiceDeConfort.setText("Légèrement tiède");
00163                 break;
00164
00165             case EspaceDeTravail.INDICE_NEUTRE:
00166                 affichageIndiceDeConfort.setText("Neutre");
00167                 break;
00168
00169             case EspaceDeTravail.INDICE_LEGEREMENT_FRAIS:
00170                 affichageIndiceDeConfort.setText("Légèrement frais");
00171                 break;
00172
00173             case EspaceDeTravail.INDICE_FRAIS:
00174                 affichageIndiceDeConfort.setText("Frais");
00175                 break;
00176
00177             case EspaceDeTravail.INDICE_FROID:
00178                 affichageIndiceDeConfort.setText("Froid");
00179                 break;
00180         }
00181
00182         Log.d(TAG, "afficherIndiceDeConfort() " + espaceDeTravail.
00183             getIndiceDeConfort());
00184     }
```

8.1.2.9 afficherLieu()

```
void com.lasalle.meeting.AffichageEspaceDeTravail.afficherLieu ( )
```

Affiche le lieu de l'espace de travail.

Définition à la ligne 103 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.getLieu\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficher\(\)](#).

```
00104     {
00105         TextView affichageLieu = (TextView)findViewById(R.id.affichageLieu);
00106         affichageLieu.setText(espaceDeTravail.getLieu());
00107
00108         Log.d(TAG, "afficherLieu() " + espaceDeTravail.getLieu());
00109     }
```

8.1.2.10 afficherNom()

```
void com.lasalle.meeting.AffichageEspaceDeTravail.afficherNom ( )
```

Affiche le nom de l'espace de travail.

Définition à la ligne 92 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.getNom\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficher\(\)](#).

```
00093     {
00094         TextView affichageNom = (TextView)findViewById(R.id.affichageNom);
00095         affichageNom.setText(espaceDeTravail.getNom\(\));
00096
00097         Log.d(TAG, "afficherNom() " + espaceDeTravail.getNom\(\));
00098     }
```

8.1.2.11 afficherSuperficie()

```
void com.lasalle.meeting.AffichageEspaceDeTravail.afficherSuperficie ( )
```

Affiche la superficie de l'espace de travail.

Définition à la ligne 125 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.getSuperficie\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficher\(\)](#).

```
00126     {
00127         TextView affichageSuperficie = (TextView)findViewById(R.id.affichageSuperficie);
00128         affichageSuperficie.setText (String.valueOf(espaceDeTravail.
00129             getSuperficie\(\)));
00129
00130         Log.d(TAG, "afficherSuperficie() " + espaceDeTravail.
00131             getSuperficie\(\));
00131     }
```

8.1.2.12 afficherTemperature()

```
void com.lasalle.meeting.AffichageEspaceDeTravail.afficherTemperature ( )
```

Affiche la température de l'espace de travail.

Définition à la ligne 136 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.getTemperature\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficher\(\)](#).

```
00137     {
00138         TextView affichageTemperature = (TextView)findViewById(R.id.affichageTemperature);
00139         affichageTemperature.setText (String.valueOf(espaceDeTravail.
00140             getTemperature\(\)));
00140
00141         Log.d(TAG, "afficherTemperature() " + espaceDeTravail.
00142             getTemperature\(\));
00142     }
```


8.1.2.13 `finish()`

```
void com.lasalle.meeting.AffichageEspaceDeTravail.finish ( )
```

Termine l'activité d'affichage d'un espace de travail.

Définition à la ligne 372 du fichier [AffichageEspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.onActivityResult\(\)](#).

```
00373     {
00374         Log.d(TAG, "finish()");
00375
00376         Intent intent = new Intent();
00377         //intent.putExtra("unEspaceDeTravail", espaceDeTravail);
00378         setResult(RESULT_OK, intent);
00379         super.finish();
00380     }
```

8.1.2.14 `onActivityResult()`

```
void com.lasalle.meeting.AffichageEspaceDeTravail.onActivityResult (
    int requestCode,
    int resultCode,
    Intent data ) [protected]
```

Traite le retour de l'activité de modification d'un espace de travail.

Définition à la ligne 360 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.AffichageEspaceDeTravail.finish\(\)](#).

```
00361     {
00362         super.onActivityResult(requestCode, resultCode, data);
00363         Log.d(TAG, "onActivityResult() requestCode=" + requestCode + " - resultCode=" + resultCode + "");
00364     };
00365     finish();
00366 }
```

8.1.2.15 `onCreate()`

```
void com.lasalle.meeting.AffichageEspaceDeTravail.onCreate (
    Bundle savedInstanceState ) [protected]
```

Méthode appelée à la création de l'activité

Définition à la ligne 50 du fichier [AffichageEspaceDeTravail.java](#).

Références [com.lasalle.meeting.AffichageEspaceDeTravail.afficher\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.handler](#), et [com.lasalle.meeting.EspaceDeTravail.initialiserCommunication\(\)](#).

```
00051     {
00052         super.onCreate(savedInstanceState);
00053         setContentView(R.layout.activity_affichage_espace_de_travail);
00054         Intent intent = getIntent();
00055         espaceDeTravail = (EspaceDeTravail)intent.getSerializableExtra("unEspaceDeTravail");
00056         espaceDeTravail.initialiserCommunication(
00057             handler);
00058         afficher();
00059     }
```

8.1.3 Documentation des données membres

8.1.3.1 espaceDeTravail

`EspaceDeTravail` `com.lasalle.meeting.AffichageEspaceDeTravail.espaceDeTravail` [private]

L'espace de travail.

Les attributs

Définition à la ligne 44 du fichier [AffichageEspaceDeTravail.java](#).

8.1.3.2 handler

`Handler` `com.lasalle.meeting.AffichageEspaceDeTravail.handler` [private]

Définition à la ligne 382 du fichier [AffichageEspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.onCreate\(\)](#).

8.1.3.3 TAG

`final String` `com.lasalle.meeting.AffichageEspaceDeTravail.TAG` = `"_AffichageEspaceTravail"` [static], [private]

TAG pour les logs.

Les constantes

Définition à la ligne 39 du fichier [AffichageEspaceDeTravail.java](#).

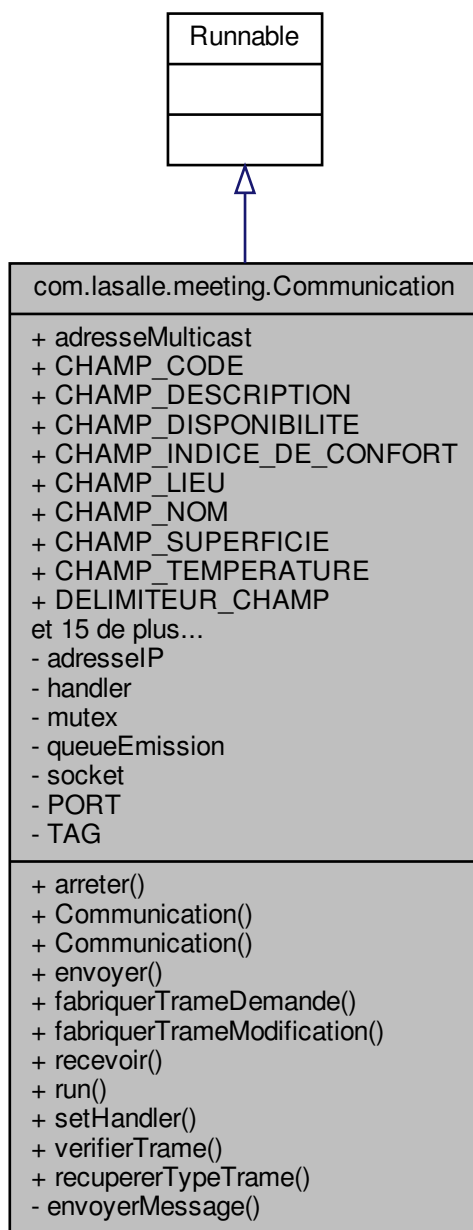
La documentation de cette classe a été générée à partir du fichier suivant :

— [AffichageEspaceDeTravail.java](#)

8.2 Référence de la classe com.lasalle.meeting.Communication

[Communication](#) entre l'application et le portier.

Graphe de collaboration de com.lasalle.meeting.Communication :



Fonctions membres publiques

- void [arreter](#) ()
Arrête la socket, donc la communication avec les portiers.
- [Communication](#) (Handler [handler](#))
Constructeur par défaut de la classe [Communication](#).
- [Communication](#) ()

- void **envoyer** (String trame, String adressePortier)
Envoyer la trame.
- String **fabriquerTrameDemande** (int typeTrame)
Fabrique la trame de demande.
- String **fabriquerTrameModification** (int typeTrame, List< String > parametres)
Fabrique la trame de modification.
- void **recevoir** ()
Recevoir les trames des portiers.
- void **run** ()
Assure la réception des trames.
- void **setHandler** (Handler handler)
- boolean **verifierTrame** (String trame)
Vérifie la trame.

Fonctions membres publiques statiques

- static int **recupererTypeTrame** (String[] champs)
Détermine le type de trame.

Attributs publics statiques

- static final String **adresseMulticast** = "239.0.0.42"
Adresse multicast des portiers.
- static final int **CHAMP_CODE** = 1
- static final int **CHAMP_DESCRIPTION** = 1
- static final int **CHAMP_DISPONIBILITE** = 4
- static final int **CHAMP_INDICE_DE_CONFORT** = 5
- static final int **CHAMP_LIEU** = 2
- static final int **CHAMP_NOM** = 0
- static final int **CHAMP_SUPERFICIE** = 3
- static final int **CHAMP_TEMPERATURE** = 6
- static final String **DELIMITEUR_CHAMP** = " ; "
- static final String **DELIMITEUR_EN_TETE** = "\$"
- static final String **DELIMITEUR_FIN** = "\r\n"
- static final int **DEMANDE_DISPONIBILITE** = 3
- static final int **DEMANDE_INFORMATIONS** = 1
- static final int **MODIFICATION_DISPONIBILITE** = 3
- static final int **MODIFICATION_INFORMATIONS** = 1
- static final int **NB_CHAMPS_DEMANDE_DISPONIBILITE** = 1
- static final int **NB_CHAMPS_DEMANDE_INFORMATIONS** = 7
- static final int **NB_CHAMPS_DISPONIBILITE** = 1
- static final int **NB_CHAMPS_DISPONIBILITE_CODE** = 2
- static final int **NB_CHAMPS_INFORMATIONS** = 4
- static final int **NB_CHAMPS_MODIFICATION_DISPONIBILITE** = 2
- static final int **NB_CHAMPS_RETOUR_MODIFICATION_DISPONIBILITE** = 3
- static final int **TRAME_INCONNUE** = -1
- static final int **TYPE_RECEPTION** = 1
Code du message indiquant une réception de données.

Fonctions membres privées

- void **envoyerMessage** (int type, String adresse, int port, String donnees)
Envoie un message.

Attributs privés

- InetAddress **adresseIP** = null
Adresse IP du portier.
- Handler **handler**
Handler permettant l'échange de Message avec l'activité
- final ReentrantLock **mutex** = new ReentrantLock()
- LinkedBlockingQueue< DatagramPacket > **queueEmission**
Queue d'émission des trames.
- DatagramSocket **socket** = null
Socket UDP.

Attributs privés statiques

- static final int `PORT` = 5000
Port d'écoute des portiers.
- static final String `TAG` = "_Communication"
TAG pour les logs.

8.2.1 Description détaillée

[Communication](#) entre l'application et le portier.

Définition à la ligne 29 du fichier [Communication.java](#).

8.2.2 Documentation des constructeurs et destructeur

8.2.2.1 `Communication()` [1/2]

```
com.lasalle.meeting.Communication.Communication (
    Handler handler )
```

Constructeur par défaut de la classe [Communication](#).

Paramètres

<i>handler</i>	Handler
----------------	---------

Définition à la ligne 79 du fichier [Communication.java](#).

Références [com.lasalle.meeting.Communication.handler](#).

```
00080     {
00081         this.handler = handler;
00082
00083         try
00084         {
00085             socket = new DatagramSocket(PORT);
00086             Log.d(TAG, "Création de la socket UDP sur le port " + PORT);
00087         }
00088         catch (Exception e)
00089         {
00090             e.printStackTrace();
00091         }
00092
00093         queueEmission = new LinkedBlockingQueue<DatagramPacket>();
00094     }
```

8.2.2.2 `Communication()` [2/2]

```
com.lasalle.meeting.Communication.Communication ( )
```

Définition à la ligne 96 du fichier [Communication.java](#).

```

00097      {
00098          this.handler = null;
00099
00100          try
00101          {
00102              socket = new DatagramSocket();
00103              Log.d(TAG, "Création d'une socket UDP");
00104          }
00105          catch (Exception e)
00106          {
00107              e.printStackTrace();
00108          }
00109
00110          queueEmission = new LinkedBlockingQueue<DatagramPacket>();
00111      }

```

8.2.3 Documentation des fonctions membres

8.2.3.1 arreter()

```
void com.lasalle.meeting.Communication.arreter ( )
```

Arrête la socket, donc la communication avec les portiers.

Définition à la ligne 331 du fichier [Communication.java](#).

Référéncé par [com.lasalle.meeting.EspaceDeTravail.initialiserCommunication\(\)](#).

```

00332      {
00333          if (socket == null)
00334              return;
00335          socket.close();
00336      }

```

8.2.3.2 envoyer()

```
void com.lasalle.meeting.Communication.envoyer (
    String trame,
    String adressePortier )
```

Envoyer la trame.

Paramètres

<i>trame</i>	la trame à envoyer
<i>adressePortier</i>	l'adresse IP du portier

Définition à la ligne 123 du fichier [Communication.java](#).

Références [com.lasalle.meeting.Communication.run\(\)](#).

Référéncé par [com.lasalle.meeting.EspaceDeTravail.demanderInformations\(\)](#), [com.lasalle.meeting.IHMMeeting.demarrerReseau\(\)](#), [com.lasalle.meeting.IHMMeeting.initialiserEspacesDeTravail\(\)](#), [com.lasalle.meeting.EspaceDeTravail.liberer\(\)](#), [com.lasalle.meeting.EspaceDeTravail.modifierInformations\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.reserver\(\)](#).

```

00124     {
00125         if(socket == null || socket.isClosed())
00126             return;
00127
00128         Log.d(TAG, "envoyer() : adressePortier = " + adressePortier);
00129
00130         final InetAddress adresseIPDistant;
00131         try
00132         {
00133             adresseIPDistant = InetAddress.getByNames(adressePortier);
00134         }
00135         catch (UnknownHostException e)
00136         {
00137             e.printStackTrace();
00138             return;
00139         }
00140
00141         // Crée et démarre un thread pour envoyer la trame
00142         new Thread()
00143         {
00144             @Override public void run()
00145             {
00146                 byte[] emission = new byte[1024];
00147
00148                 try
00149                 {
00150                     emission = trame.getBytes();
00151                     DatagramPacket paquetRetour = new DatagramPacket(emission, emission.length,
00152                         adresseIPDistant, PORT);
00153                     socket.send(paquetRetour);
00154                     Log.d(TAG, "envoyer() : " + trame + " à " + adresseIPDistant + ":" +
00155                         PORT);
00156                 }
00157                 catch (IOException e)
00158                 {
00159                     e.printStackTrace();
00160                     Log.d(TAG, "Erreur émission !");
00161                 }
00162             }
00163         }.start();
00164     }

```

8.2.3.3 envoyerMessage()

```

void com.lasalle.meeting.Communication.envoyerMessage (
    int type,
    String adresse,
    int port,
    String donnees ) [private]

```

Envoie un message.

Définition à la ligne 199 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.Communication.recevoir\(\)](#).

```

00200     {
00201         if(handler == null)
00202             return;
00203         Message msg = Message.obtain();
00204         msg.what = type;
00205         Bundle b = new Bundle();
00206         b.putString("adresseIP", adresse);
00207         b.putInt("port", port);
00208         b.putString("donnees", donnees);
00209         msg.setData(b);
00210         mutex.lock();
00211         handler.sendMessage(msg);
00212         mutex.unlock();
00213         Log.d(TAG, "envoyerMessage() -> handler.sendMessage()");
00214     }

```

8.2.3.4 fabriquerTrameDemande()

```

String com.lasalle.meeting.Communication.fabriquerTrameDemande (
    int typeTrame )

```

Fabrique la trame de demande.

Paramètres

<i>typeTrame</i>	le type de trame de demande
------------------	-----------------------------

Renvoi

trame la trame fabriquée

Définition à la ligne 221 du fichier [Communication.java](#).

Références [com.lasalle.meeting.Communication.DELIMITEUR_FIN](#), [com.lasalle.meeting.Communication.DEMANDE_DISPONIBILITE](#), et [com.lasalle.meeting.Communication.DEMANDE_INFORMATIONS](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.demanderInformations\(\)](#), [com.lasalle.meeting.IHMMeeting.demarrerReseau\(\)](#), et [com.lasalle.meeting.IHMMeeting.initialiserEspacesDeTravail\(\)](#).

```

00222     {
00223         /*
00224          * Protocole
00225          *
00226          * Demande informations du portier :
00227          * $GET;1\r\n
00228          *
00229          * Demande disponibilité du portier :
00230          * $GET;3\r\n
00231          */
00232
00233         String trame = null;
00234         Log.d(TAG, "fabriquerTrameDemande() type = " + typeTrame);
00235
00236         switch(typeTrame)
00237         {
00238             case DEMANDE_INFORMATIONS:
00239                 trame = DELIMITEUR_EN_TETE + "GET;1" +
00240                     DELIMITEUR_FIN;
00241                 break;
00242             case DEMANDE_DISPONIBILITE:
00243                 trame = DELIMITEUR_EN_TETE + "GET;3" +
00244                     DELIMITEUR_FIN;
00245                 break;
00246             default:
00247                 Log.d(TAG, "fabriquerTrameDemande() : type de trame inconnu !");
00248         }
00249         Log.d(TAG, "fabriquerTrameDemande() trame = " + trame);
00250         return trame;
00251     }

```

8.2.3.5 fabriquerTrameModification()

```

String com.lasalle.meeting.Communication.fabriquerTrameModification (
    int typeTrame,
    List< String > parametres )

```

Fabrique la trame de modification.

Paramètres

<i>typeTrame</i>	le type de trame de modification
------------------	----------------------------------

Renvoi

trame la trame fabriquée

Définition à la ligne 258 du fichier [Communication.java](#).

Références [com.lasalle.meeting.Communication.DELIMITEUR_FIN](#), [com.lasalle.meeting.Communication.MODIFICATION_DISPONIBILITE](#), [com.lasalle.meeting.Communication.MODIFICATION_INFORMATIONS](#), et [com.lasalle.meeting.Communication.NB_CHAMPS_INFORMATIONS](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.liberer\(\)](#), [com.lasalle.meeting.EspaceDeTravail.modifierInformations\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.reserver\(\)](#).

```

00259     {
00260         /*
00261          * Protocole
00262          *
00263          * Actualiser les informations d'un portier :
00264          * $SET;1;nomSalle;description;emplacement;surface\r\n
00265          *
00266          * Actualiser la disponibilité d'un portier :
00267          * $SET;3;disponibilité\r\n
00268          *
00269          * Exemple d'utilisation :
00270          * List<String> parametres = Arrays.asList("B11", "Salle de cours", "Batiment BTS", "25");
00271          * String trame =
communication.fabriquerTrameModification(Communication.MODIFICATION_INFORMATIONS, parametres);
00272          */
00273
00274         // Vérifications
00275         if(parametres == null)
00276             return null;
00277
00278         if(parametres.size() < 1)
00279             return null;
00280
00281         Log.d(TAG, "fabriquerTrameModification() type = " + typeTrame);
00282         Log.d(TAG, "fabriquerTrameModification() Nb parametres = " + parametres.size());
00283         for(int i=0;i<parametres.size();++i)
00284         {
00285             Log.d(TAG, "fabriquerTrameModification() parametres[" + i + "] = " + parametres.get(i));
00286         }
00287
00288         String trame = null;
00289
00290         switch(typeTrame)
00291         {
00292             case MODIFICATION_INFORMATIONS:
00293                 if(parametres.size() != NB_CHAMPS_INFORMATIONS)
00294                     return null;
00295                 trame = DELIMITEUR_EN_TETE + "SET;1;" + parametres.get(
CHAMP_NOM) + DELIMITEUR_CHAMP + parametres.get(
CHAMP_DESCRIPTION) + DELIMITEUR_CHAMP + parametres.get(
CHAMP_LIEU) + DELIMITEUR_CHAMP + parametres.get(
CHAMP_SUPERFICIE) + DELIMITEUR_FIN;
00296                 break;
00297
00298             case MODIFICATION_DISPONIBILITE:
00299                 if(parametres.size() >= NB_CHAMPS_DISPONIBILITE && parametres.size()
<= (NB_CHAMPS_DISPONIBILITE+1))
00300                 {
00301                     if (parametres.get(0).equals("0"))
00302                         trame = DELIMITEUR_EN_TETE + "SET;3;" + parametres.get(0) +
DELIMITEUR_FIN;
00303                     else
00304                         trame = DELIMITEUR_EN_TETE + "SET;3;" + parametres.get(0) + ";" +
parametres.get(1) + DELIMITEUR_FIN;
00305                 }
00306                 break;
00307
00308             default:
00309                 Log.d(TAG, "fabriquerTrameModification() : type de trame inconnu !");
00310         }
00311
00312         Log.d(TAG, "fabriquerTrameModification() trame = " + trame);
00313
00314         return trame;
00315     }

```

8.2.3.6 recevoir()

```
void com.lasalle.meeting.Communication.recevoir ( )
```

Recevoir les trames des portiers.

Définition à la ligne 167 du fichier [Communication.java](#).

Références [com.lasalle.meeting.Communication.envoyerMessage\(\)](#), et [com.lasalle.meeting.Communication.verifierTrame\(\)](#).

Référencé par [com.lasalle.meeting.Communication.run\(\)](#).

```
00168     {
00169         byte[] reception = new byte[1024];
00170
00171         while (socket != null && !socket.isClosed())
00172         {
00173             try
00174             {
00175                 final DatagramPacket paquetRecu = new DatagramPacket(reception, reception.length);
00176                 socket.receive(paquetRecu);
00177                 final String donnees = new String(paquetRecu.getData(), paquetRecu.getOffset(), paquetRecu.
00178                     getLength());
00179                 Log.d(TAG, "Réception [" + paquetRecu.getAddress().getHostAddress() + ":" + paquetRecu.
00180                     getPort() + "] -> " + donnees);
00181                 if(verifierTrame(donnees))
00182                 {
00183                     // Envoie les données reçues à l'activité
00184                     envoyerMessage(TYPE_RECEPTION, paquetRecu.getAddress().
00185                         getHostAddress(), paquetRecu.getPort(), donnees);
00186                 }
00187             }
00188             catch (Exception e)
00189             {
00190                 e.printStackTrace();
00191                 Log.d(TAG, "Erreur réception !");
00192             }
00193             Log.d(TAG, "recevoir()");
00194         }
```

8.2.3.7 recupererTypeTrame()

```
static int com.lasalle.meeting.Communication.recupererTypeTrame (
    String [] champs ) [static]
```

Détermine le type de trame.

Paramètres

<i>champs</i>	tableau contenant les champs de la trame
---------------	--

Renvoie

typeTrame le type de trame

Définition à la ligne 353 du fichier [Communication.java](#).

Références [com.lasalle.meeting.Communication.DEMANDE_DISPONIBILITE](#), [com.lasalle.meeting.Communication.DEMANDE_](#)↵
[INFORMATIONS](#), [com.lasalle.meeting.Communication.MODIFICATION_DISPONIBILITE](#), [com.lasalle.meeting.Communication.](#)↵
[NB_CHAMPS_DEMANDE_DISPONIBILITE](#), [com.lasalle.meeting.Communication.NB_CHAMPS_DEMANDE_INFORMATIO](#)↵
[NS](#), [com.lasalle.meeting.Communication.NB_CHAMPS_RETOUR_MODIFICATION_DISPONIBILITE](#), et [com.lasalle.meeting.](#)↵
[Communication.TRAME_INCONNUE](#).

```

00354     {
00355         int typeTrame = Communication.TRAME_INCONNUE;
00356
00357         switch(champs.length)
00358         {
00359             case Communication.NB_CHAMPS_DEMANDE_INFORMATIONS:
00360                 Log.d(TAG, "handleMessage() Trame DEMANDE_INFORMATIONS");
00361                 typeTrame = Communication.DEMANDE_INFORMATIONS;
00362                 break;
00363             case Communication.NB_CHAMPS_DEMANDE_DISPONIBILITE:
00364                 Log.d(TAG, "handleMessage() Trame DEMANDE_DISPONIBILITE");
00365                 typeTrame = Communication.DEMANDE_DISPONIBILITE;
00366                 break;
00367             case Communication.NB_CHAMPS_RETOUR_MODIFICATION_DISPONIBILITE:
00368                 Log.d(TAG, "handleMessage() Trame MODIFICATION_DISPONIBILITE");
00369                 typeTrame = Communication.MODIFICATION_DISPONIBILITE;
00370         }
00371
00372         return typeTrame;
00373     }

```

8.2.3.8 `run()`

```
void com.lasalle.meeting.Communication.run ( )
```

Assure la réception des trames.

Définition à la ligne 342 du fichier [Communication.java](#).

Références [com.lasalle.meeting.Communication.recevoir\(\)](#).

Référencé par [com.lasalle.meeting.Communication.envoyer\(\)](#).

```

00343     {
00344         Log.d(TAG, "Démarre le thread réception");
00345         recevoir();
00346     }

```

8.2.3.9 `setHandler()`

```
void com.lasalle.meeting.Communication.setHandler (
    Handler handler )
```

Définition à la ligne 113 du fichier [Communication.java](#).

Références [com.lasalle.meeting.Communication.handler](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.initialiserCommunication\(\)](#).

```

00114     {
00115         this.handler = handler;
00116     }

```

8.2.3.10 `verifierTrame()`

```
boolean com.lasalle.meeting.Communication.verifierTrame (
    String trame )
```

Vérifie la trame.

Paramètres

<i>trame</i>	la trame à vérifier
--------------	---------------------

Définition à la ligne 321 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.Communication.recevoir\(\)](#).

```
00322     {
00323         Log.d(TAG, "verifierTrame() " + (trame.startsWith(DELIMITEUR_EN_TETE) && trame
00324             .endsWith(DELIMITEUR_FIN)));
00325         return (trame.startsWith(DELIMITEUR_EN_TETE) && trame.endsWith(
00326             DELIMITEUR_FIN));
00327     }
```

8.2.4 Documentation des données membres

8.2.4.1 adresseIP

```
InetAddress com.lasalle.meeting.Communication.adresseIP = null [private]
```

Adresse IP du portier.

Les attributs

Définition à la ligne 39 du fichier [Communication.java](#).

8.2.4.2 adresseMulticast

```
final String com.lasalle.meeting.Communication.adresseMulticast = "239.0.0.42" [static]
```

Adresse multicast des portiers.

Définition à la ligne 40 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.IHMMeeting.demarrerReseau\(\)](#), et [com.lasalle.meeting.IHMMeeting.initialiserEspacesDeTravail\(\)](#).

8.2.4.3 CHAMP_CODE

```
final int com.lasalle.meeting.Communication.CHAMP_CODE = 1 [static]
```

Définition à la ligne 73 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.extraireCode\(\)](#).

8.2.4.4 CHAMP_DESCRIPTION

```
final int com.lasalle.meeting.Communication.CHAMP_DESCRIPTION = 1 [static]
```

Définition à la ligne 67 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.extraireInformations\(\)](#).

8.2.4.5 CHAMP_DISPONIBILITE

```
final int com.lasalle.meeting.Communication.CHAMP_DISPONIBILITE = 4 [static]
```

Définition à la ligne 70 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.extraireInformations\(\)](#).

8.2.4.6 CHAMP_INDICE_DE_CONFORT

```
final int com.lasalle.meeting.Communication.CHAMP_INDICE_DE_CONFORT = 5 [static]
```

Définition à la ligne 71 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.extraireInformations\(\)](#).

8.2.4.7 CHAMP_LIEU

```
final int com.lasalle.meeting.Communication.CHAMP_LIEU = 2 [static]
```

Définition à la ligne 68 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.extraireInformations\(\)](#).

8.2.4.8 CHAMP_NOM

```
final int com.lasalle.meeting.Communication.CHAMP_NOM = 0 [static]
```

Définition à la ligne 66 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.extraireCode\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.extraireInformations\(\)](#).

8.2.4.9 CHAMP_SUPERFICIE

```
final int com.lasalle.meeting.Communication.CHAMP_SUPERFICIE = 3 [static]
```

Définition à la ligne 69 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.extraireInformations\(\)](#).

8.2.4.10 CHAMP_TEMPERATURE

```
final int com.lasalle.meeting.Communication.CHAMP_TEMPERATURE = 6 [static]
```

Définition à la ligne 72 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.extraireInformations\(\)](#).

8.2.4.11 DELIMITEUR_CHAMP

```
final String com.lasalle.meeting.Communication.DELIMITEUR_CHAMP = ";" [static]
```

Définition à la ligne 52 du fichier [Communication.java](#).

8.2.4.12 DELIMITEUR_EN_TETE

```
final String com.lasalle.meeting.Communication.DELIMITEUR_EN_TETE = "$" [static]
```

Protocole

Définition à la ligne 51 du fichier [Communication.java](#).

8.2.4.13 DELIMITEUR_FIN

```
final String com.lasalle.meeting.Communication.DELIMITEUR_FIN = "\r\n" [static]
```

Définition à la ligne 53 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.Communication.fabriquerTrameDemande\(\)](#), et [com.lasalle.meeting.Communication.fabriquerTrameModification\(\)](#).

8.2.4.14 DEMANDE_DISPONIBILITE

```
final int com.lasalle.meeting.Communication.DEMANDE_DISPONIBILITE = 3 [static]
```

Définition à la ligne 56 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.Communication.fabriquerTrameDemande\(\)](#), et [com.lasalle.meeting.Communication.recupererTypeTrame\(\)](#).

8.2.4.15 DEMANDE_INFORMATIONS

```
final int com.lasalle.meeting.Communication.DEMANDE_INFORMATIONS = 1 [static]
```

Définition à la ligne 55 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.demanderInformations\(\)](#), [com.lasalle.meeting.IHMMeeting.demarrerReseau\(\)](#), [com.lasalle.meeting.Communication.fabriquerTrameDemande\(\)](#), [com.lasalle.meeting.IHMMeeting.initialiserEspacesDeTravail\(\)](#), et [com.lasalle.meeting.Communication.recupererTypeTrame\(\)](#).

8.2.4.16 handler

```
Handler com.lasalle.meeting.Communication.handler [private]
```

Handler permettant l'échange de Message avec l'activité

Définition à la ligne 46 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.Communication.Communication\(\)](#), et [com.lasalle.meeting.Communication.setHandler\(\)](#).

8.2.4.17 MODIFICATION_DISPONIBILITE

```
final int com.lasalle.meeting.Communication.MODIFICATION_DISPONIBILITE = 3 [static]
```

Définition à la ligne 58 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.Communication.fabriquerTrameModification\(\)](#), [com.lasalle.meeting.EspaceDeTravail.liberer\(\)](#), [com.lasalle.meeting.Communication.recupererTypeTrame\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.reserver\(\)](#).

8.2.4.18 MODIFICATION_INFORMATIONS

```
final int com.lasalle.meeting.Communication.MODIFICATION_INFORMATIONS = 1 [static]
```

Définition à la ligne 57 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.Communication.fabriquerTrameModification\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.modifier↵ Informations\(\)](#).

8.2.4.19 mutex

```
final ReentrantLock com.lasalle.meeting.Communication.mutex = new ReentrantLock() [private]
```

Définition à la ligne 43 du fichier [Communication.java](#).

8.2.4.20 NB_CHAMPS_DEMANDE_DISPONIBILITE

```
final int com.lasalle.meeting.Communication.NB_CHAMPS_DEMANDE_DISPONIBILITE = 1 [static]
```

Définition à la ligne 63 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.Communication.recupererTypeTrame\(\)](#).

8.2.4.21 NB_CHAMPS_DEMANDE_INFORMATIONS

```
final int com.lasalle.meeting.Communication.NB_CHAMPS_DEMANDE_INFORMATIONS = 7 [static]
```

Définition à la ligne 62 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.extraireInformations\(\)](#), et [com.lasalle.meeting.Communication.recupererTypeTrame\(\)](#).

8.2.4.22 NB_CHAMPS_DISPONIBILITE

```
final int com.lasalle.meeting.Communication.NB_CHAMPS_DISPONIBILITE = 1 [static]
```

Définition à la ligne 60 du fichier [Communication.java](#).

8.2.4.23 NB_CHAMPS_DISPONIBILITE_CODE

```
final int com.lasalle.meeting.Communication.NB_CHAMPS_DISPONIBILITE_CODE = 2 [static]
```

Définition à la ligne 61 du fichier [Communication.java](#).

8.2.4.24 NB_CHAMPS_INFORMATIONS

```
final int com.lasalle.meeting.Communication.NB_CHAMPS_INFORMATIONS = 4 [static]
```

Définition à la ligne 59 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.Communication.fabriquerTrameModification\(\)](#).

8.2.4.25 NB_CHAMPS_MODIFICATION_DISPONIBILITE

```
final int com.lasalle.meeting.Communication.NB_CHAMPS_MODIFICATION_DISPONIBILITE = 2 [static]
```

Définition à la ligne 64 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.extraireCode\(\)](#).

8.2.4.26 NB_CHAMPS_RETOUR_MODIFICATION_DISPONIBILITE

```
final int com.lasalle.meeting.Communication.NB_CHAMPS_RETOUR_MODIFICATION_DISPONIBILITE = 3 [static]
```

Définition à la ligne 65 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.Communication.recupererTypeTrame\(\)](#).

8.2.4.27 PORT

```
final int com.lasalle.meeting.Communication.PORT = 5000 [static], [private]
```

Port d'écoute des portiers.

Définition à la ligne 41 du fichier [Communication.java](#).

8.2.4.28 queueEmission

```
LinkedBlockingQueue<DatagramPacket> com.lasalle.meeting.Communication.queueEmission [private]
```

Queue d'émission des trames.

Définition à la ligne 45 du fichier [Communication.java](#).

8.2.4.29 socket

```
DatagramSocket com.lasalle.meeting.Communication.socket = null [private]
```

Socket UDP.

Définition à la ligne 44 du fichier [Communication.java](#).

8.2.4.30 TAG

```
final String com.lasalle.meeting.Communication.TAG = "_Communication" [static], [private]
```

TAG pour les logs.

Les constantes

Définition à la ligne 34 du fichier [Communication.java](#).

8.2.4.31 TRAME_INCONNUE

```
final int com.lasalle.meeting.Communication.TRAME_INCONNUE = -1 [static]
```

Définition à la ligne 54 du fichier [Communication.java](#).

Référencé par [com.lasalle.meeting.Communication.recupererTypeTrame\(\)](#).

8.2.4.32 TYPE_RECEPTION

```
final int com.lasalle.meeting.Communication.TYPE_RECEPTION = 1 [static]
```

Code du message indiquant une réception de données.

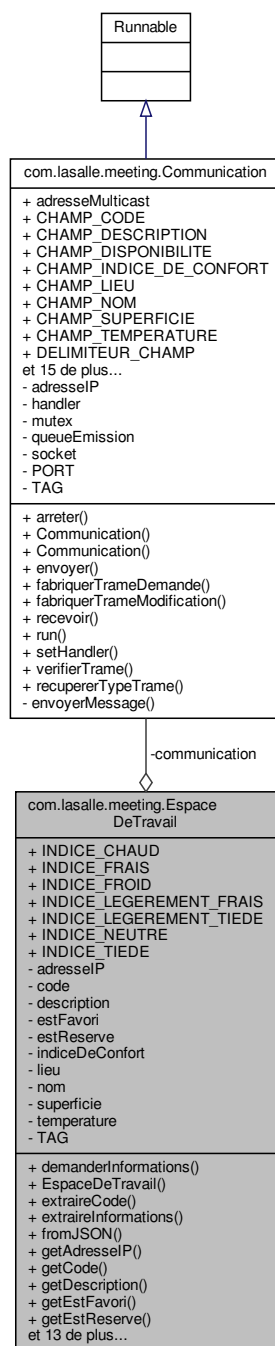
Définition à la ligne 42 du fichier [Communication.java](#).

La [documentation de cette classe](#) a été générée à partir du fichier suivant :

8.3 Référence de la classe com.lasalle.meeting.EspaceDeTravail

L'espace de travail.

Graphe de collaboration de com.lasalle.meeting.EspaceDeTravail :



Fonctions membres publiques

- void `demanderInformations` ()
Envoie une trame de demande d'informations au portier si clic sur bouton rafraichir.
- `EspaceDeTravail` (String `adresseIP`)
Constructeur par défaut de la classe `EspaceDeTravail`.
- boolean `extraireCode` (String `trame`)
Extrait le code.
- boolean `extraireInformations` (String `trame`)
Extrait les informations d'une trame `DEMANDE_INFORMATIONS`.
- void `fromJSON` (String `strJSON`)
Récupération de données JSON.
- String `getAdresseIP` ()
Accesseur de l'attribut `adresseIP`.
- String `getCode` ()
Accesseur de l'attribut `code`.
- String `getDescription` ()
Accesseur de l'attribut `description`.
- boolean `getEstFavori` ()
Accesseur de l'attribut `estFavori`.
- boolean `getEstReserve` ()
Accesseur de l'attribut `estReserve`.
- int `getIndiceDeConfort` ()
Accesseur de l'attribut `indiceDeConfort`.
- String `getLieu` ()
Accesseur de l'attribut `lieu`.
- String `getNom` ()
Accesseur de l'attribut `nom`.
- int `getSuperficie` ()
Accesseur de l'attribut `superficie`.
- double `getTemperature` ()
Accesseur de l'attribut `temperature`.
- void `initialiserCommunication` (Handler `handler`)
Initialise une communication.
- void `liberer` (String `code`)
Libère l'espace de travail.
- void `modifierInformations` (List< String > `parametres`)
Modifie les informations de l'espace de travail.
- void `reserver` ()
Réserve l'espace de travail.
- void `setCode` (String `code`)
Mutateur de l'attribut `code`.
- void `setEstFavori` (boolean `estFavori`)
Mutateur de l'attribut `estFavori`.
- void `setEstReserve` (boolean `estReserve`)
Mutateur de l'attribut `estReserve`.
- String `toJSON` ()
Création de données JSON.

Attributs publics statiques

- static final int `INDICE_CHAUD` = 3
- static final int `INDICE_FRAIS` = -2
- static final int `INDICE_FROID` = -3
- static final int `INDICE_LEGEREMENT_FRAIS` = -1
- static final int `INDICE_LEGEREMENT_TIEDE` = 1
- static final int `INDICE_NEUTRE` = 0
- static final int `INDICE_TIEDE` = 2

Attributs privés

- String `adresseIP`
Adresse IP du portier.
- String `code`
Code pour libérer l'espace de travail.
- `Communication` `communication` = null
Attribut permettant d'envoyer des requêtes.
- String `description`

- *Description de l'espace de travail.*
- boolean `estFavori`
- *Si l'espace de travail est en favori.*
- boolean `estReserve`
- *Disponibilité de l'espace de travail.*
- int `indiceDeConfort`
- *Indice de confort de l'espace de travail.*
- String `lieu`
- *Lieu de l'espace de travail.*
- String `nom`
- *Nom de l'espace de travail.*
- int `superficie`
- *Superficie de l'espace de travail.*
- double `temperature`
- *Température de l'espace de travail.*

Attributs privés statiques

- static final String `TAG` = `"_EspaceDeTravail"`
TAG pour les logs.

8.3.1 Description détaillée

L'espace de travail.

Définition à la ligne 27 du fichier `EspaceDeTravail.java`.

8.3.2 Documentation des constructeurs et destructeur

8.3.2.1 EspaceDeTravail()

```
com.lasalle.meeting.EspaceDeTravail.EspaceDeTravail (
    String adresseIP )
```

Constructeur par défaut de la classe `EspaceDeTravail`.

Définition à la ligne 59 du fichier `EspaceDeTravail.java`.

Références `com.lasalle.meeting.EspaceDeTravail.adresseIP`, `com.lasalle.meeting.EspaceDeTravail.fromJSON()`, et `com.lasalle.meeting.IHMMeeting.recupererDonneesEspaceDeTravail()`.

```
00060    {
00061        this.adresseIP = adresseIP;
00062        this.nom = "";
00063        this.lieu = "";
00064        this.description = "";
00065        this.superficie = 0;
00066        this.temperature = 0.;
00067        this.indiceDeConfort = 0;
00068        this.estReserve = false;
00069        this.code = "";
00070        this.estFavori = false;
00071
00072        fromJSON(IHMMeeting.recupererDonneesEspaceDeTravail(this));
00073    }
```

8.3.3 Documentation des fonctions membres

8.3.3.1 `demanderInformations()`

```
void com.lasalle.meeting.EspaceDeTravail.demanderInformations ( )
```

Envoie une trame de demande d'informations au portier si clic sur bouton rafraîchir.

Définition à la ligne 358 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.adresseIP](#), [com.lasalle.meeting.Communication.DEMANDE_INFORMATIONS](#), [com.lasalle.meeting.Communication.envoyer\(\)](#), et [com.lasalle.meeting.Communication.fabriquerTrameDemande\(\)](#).

```
00359     {
00360         communication.envoyer(communication.
fabriquerTrameDemande(Communication.DEMANDE_INFORMATIONS),
adresseIP);
00361     }
```

8.3.3.2 `extraireCode()`

```
boolean com.lasalle.meeting.EspaceDeTravail.extraireCode (
    String trame )
```

Extrait le code.

Paramètres

<i>trame</i>	la trame à décoder
--------------	--------------------

Protocole : `$nom;code;message`

Définition à la ligne 303 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.Communication.CHAMP_CODE](#), [com.lasalle.meeting.Communication.CHAMP_NOM](#), et [com.lasalle.meeting.Communication.NB_CHAMPS_MODIFICATION_DISPONIBILITE](#).

```
00304     {
00310         trame = trame.replace("$", "");
00311         trame = trame.replace("\r\n", "");
00312         String[] champs = trame.split(";");
00313
00314         if(champs.length == Communication.NB_CHAMPS_MODIFICATION_DISPONIBILITE)
00315         {
00316             this.nom = champs[Communication.CHAMP_NOM];
00317
00318             if (!champs[1].isEmpty())
00319             {
00320                 this.code = champs[Communication.CHAMP_CODE];
00321             }
00322
00323             Log.d(TAG, "extraireCode() nom : " + nom + " - code : " + code);
00324
00325             return true;
00326         }
00327
00328         return false;
00329     }
```

8.3.3.3 `extraireInformations()`

```
boolean com.lasalle.meeting.EspaceDeTravail.extraireInformations (
    String trame )
```

Extrait les informations d'une trame `DEMANDE_INFORMATIONS`.

Paramètres

<i>trame</i>	la trame à décoder
--------------	--------------------

Protocole : \$nom ;description ;lieu ;superficie ;disponibilité ;niveauDeConfort ;température

Définition à la ligne 251 du fichier `EspaceDeTravail.java`.

Références `com.lasalle.meeting.Communication.CHAMP_DESCRIPTION`, `com.lasalle.meeting.Communication.CHAMP_DISPONIBILITE`, `com.lasalle.meeting.Communication.CHAMP_INDICE_DE_CONFORT`, `com.lasalle.meeting.Communication.CHAMP_LIEU`, `com.lasalle.meeting.Communication.CHAMP_NOM`, `com.lasalle.meeting.Communication.CHAMP_SUPERFICIE`, `com.lasalle.meeting.Communication.CHAMP_TEMPERATURE`, et `com.lasalle.meeting.Communication.NB_CHAMPS_DEMANDE_INFORMATIONS`.

```

00252     {
00258         trame = trame.replace("$", "");
00259         trame = trame.replace("\r\n", "");
00260         String[] champs = trame.split(";");
00261
00262         if(champs.length == Communication.NB_CHAMPS_DEMANDE_INFORMATIONS)
00263         {
00264             this.nom = champs[Communication.CHAMP_NOM];
00265             this.description = champs[Communication.CHAMP_DESCRIPTION];
00266             this.lieu = champs[Communication.CHAMP_LIEU];
00267             if (!champs[Communication.CHAMP_SUPERFICIE].isEmpty())
00268             {
00269                 this.superficie = Integer.parseInt(champs[Communication.CHAMP_SUPERFICIE]);
00270             }
00271             if (!champs[Communication.CHAMP_DISPONIBILITE].isEmpty())
00272             {
00273                 if (Integer.parseInt(champs[Communication.CHAMP_DISPONIBILITE]) == 1)
00274                 {
00275                     this.estReserve = false;
00276                 }
00277                 else
00278                 {
00279                     this.estReserve = true;
00280                 }
00281             }
00282             if (!champs[Communication.CHAMP_TEMPERATURE].isEmpty())
00283             {
00284                 this.temperature = Double.parseDouble(champs[Communication.CHAMP_TEMPERATURE]);
00285             }
00286             if (!champs[Communication.CHAMP_INDICE_DE_CONFORT].isEmpty())
00287             {
00288                 this.indiceDeConfort = Integer.parseInt(champs[Communication.
CHAMP_INDICE_DE_CONFORT]);
00289             }
00290
00291             Log.d(TAG, "extraireInformations() nom : " + nom + " - description : " +
description + " - lieu : " + lieu + " - superficie : " +
superficie + " - estReserve : " + estReserve + " - temperature : " +
temperature + " - indiceDeConfort : " + indiceDeConfort);
00292
00293             return true;
00294         }
00295
00296         return false;
00297     }

```

8.3.3.4 fromJSON()

```

void com.lasalle.meeting.EspaceDeTravail.fromJSON (
    String strJSON )

```

Récupération de données JSON.

Paramètres

strJSON	les données (code et favori) formatés en JSON à extraire
----------------	--

Définition à la ligne 389 du fichier [EspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.EspaceDeTravail\(\)](#).

```

00390     {
00391         try
00392         {
00393             //Log.i(TAG, "fromJSON() JSON = " + strJSON);
00394             JSONObject json = new JSONObject(strJSON);
00395
00396             this.code = json.getString("code");
00397             this.estFavori = json.getBoolean("estFavori");
00398         }
00399         catch (JSONException e)
00400         {
00401             e.printStackTrace();
00402             Log.i(TAG, "fromJSON() Erreur !");
00403         }
00404     }

```

8.3.3.5 getAddressIP()

String com.lasalle.meeting.EspaceDeTravail.getAddressIP ()

Accesseur de l'attribut adresseIP.

Renvoie

adresseIP adresse IP de l'espace de travail

Définition à la ligne 79 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.adresseIP](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherAdresseIP\(\)](#), [com.lasalle.meeting.EspaceDeTravail.initialiser←Communication\(\)](#), et [com.lasalle.meeting.IHMMeeting.trierEspacesDeTravail\(\)](#).

```

00080     {
00081         return adresseIP;
00082     }

```

8.3.3.6 getCode()

String com.lasalle.meeting.EspaceDeTravail.getCode ()

Accesseur de l'attribut code.

Renvoie

code code pour libérer l'espace de travail

Définition à la ligne 151 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.code](#).

Référencé par [com.lasalle.meeting.IHMMeeting.sauvegarderDonneesEspaceDeTravail\(\)](#).

```

00152     {
00153         return code;
00154     }

```

8.3.3.7 getDescription()

```
String com.lasalle.meeting.EspaceDeTravail.getDescription ( )
```

Accesseur de l'attribut description.

Renvoie

description description de l'espace de travail

Définition à la ligne 106 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.description](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherDescription\(\)](#), [com.lasalle.meeting.ModificationEspaceDeTravail.afficherEditionDescription\(\)](#), et [com.lasalle.meeting.EspaceDeTravailAdaptateur.getView\(\)](#).

```
00107    {  
00108        return description;  
00109    }
```

8.3.3.8 getEstFavori()

```
boolean com.lasalle.meeting.EspaceDeTravail.getEstFavori ( )
```

Accesseur de l'attribut estFavori.

Renvoie

estFavori si l'espace de travail est dans les favoris

Définition à la ligne 160 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.estFavori](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherBoutons\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherFavori\(\)](#), [com.lasalle.meeting.EspaceDeTravailAdaptateur.getView\(\)](#), et [com.lasalle.meeting.IHMMeeting.sauvegarderDonneesEspaceDeTravail\(\)](#).

```
00161    {  
00162        return estFavori;  
00163    }
```

8.3.3.9 getEstReserve()

```
boolean com.lasalle.meeting.EspaceDeTravail.getEstReserve ( )
```

Accesseur de l'attribut estReserve.

Renvoie

estReserve disponibilité de l'espace de travail

Définition à la ligne 142 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.estReserve](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherBoutons\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherDisponibilite\(\)](#), et [com.lasalle.meeting.EspaceDeTravailAdaptateur.getView\(\)](#).

```
00143    {  
00144        return estReserve;  
00145    }
```


8.3.3.10 `getIndiceDeConfort()`

```
int com.lasalle.meeting.EspaceDeTravail.getIndiceDeConfort ( )
```

Accesseur de l'attribut `indiceDeConfort`.

Renvoie

`indiceDeConfort` indice de confort de l'espace de travail

Définition à la ligne 133 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.indiceDeConfort](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherIndiceDeConfort\(\)](#), et [com.lasalle.meeting.EspaceDeTravailAdaptateur.getView\(\)](#).

```
00134     {  
00135         return indiceDeConfort;  
00136     }
```

8.3.3.11 `getLieu()`

```
String com.lasalle.meeting.EspaceDeTravail.getLieu ( )
```

Accesseur de l'attribut `lieu`.

Renvoie

`lieu` lieu de l'espace de travail

Définition à la ligne 97 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.lieu](#).

Référencé par [com.lasalle.meeting.ModificationEspaceDeTravail.afficherEditionLieu\(\)](#), et [com.lasalle.meeting.AffichageEspaceDeTravail.afficherLieu\(\)](#).

```
00098     {  
00099         return lieu;  
00100     }
```

8.3.3.12 `getNom()`

```
String com.lasalle.meeting.EspaceDeTravail.getNom ( )
```

Accesseur de l'attribut `nom`.

Renvoie

`nom` nom de l'espace de travail

Définition à la ligne 88 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.nom](#).

Référencé par [com.lasalle.meeting.ModificationEspaceDeTravail.afficherEditionNom\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherNom\(\)](#), [com.lasalle.meeting.EspaceDeTravailAdaptateur.getView\(\)](#), [com.lasalle.meeting.IHMMeeting.trierEspacesDeTravail\(\)](#), et [com.lasalle.meeting.IHMMeeting.verifierPresenceEspaceDeTravail\(\)](#).

```
00089     {  
00090         return nom;  
00091     }
```

8.3.3.13 getSuperficie()

```
int com.lasalle.meeting.EspaceDeTravail.getSuperficie ( )
```

Accesseur de l'attribut superficie.

Renvoie

superficie superficie de l'espace de travail

Définition à la ligne 115 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.superficie](#).

Référencé par [com.lasalle.meeting.ModificationEspaceDeTravail.afficherEditionSuperficie\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.afficherSuperficie\(\)](#), et [com.lasalle.meeting.IHMMeeting.trierEspacesDeTravail\(\)](#).

```
00116    {
00117        return superficie;
00118    }
```

8.3.3.14 getTemperature()

```
double com.lasalle.meeting.EspaceDeTravail.getTemperature ( )
```

Accesseur de l'attribut temperature.

Renvoie

temperature température de l'espace de travail

Définition à la ligne 124 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.temperature](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherTemperature\(\)](#).

```
00125    {
00126        return temperature;
00127    }
```

8.3.3.15 initialiserCommunication()

```
void com.lasalle.meeting.EspaceDeTravail.initialiserCommunication (
    Handler handler )
```

Initialise une communication.

Paramètres

<i>handler</i>	le handler pour échanger des messages avec l'activité
----------------	---

Définition à la ligne 335 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.Communication.arreter\(\)](#), [com.lasalle.meeting.EspaceDeTravail.getAdresseIP\(\)](#), et [com.lasalle.meeting.Communication.setHandler\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherBoutons\(\)](#), [com.lasalle.meeting.AffichageEspaceDeTravail.onCreate\(\)](#), et [com.lasalle.meeting.ModificationEspaceDeTravail.onCreate\(\)](#).

```

00336     {
00337         Log.d(TAG, "initialiserCommunication()");
00338         if(communication == null)
00339         {
00340             communication = new Communication();
00341             communication.setHandler(handler);
00342
00343             // Démarre la réception des trames des portiers
00344             Thread tCommunicationUDP = new Thread(communication,
getAdresseIP());
00345             tCommunicationUDP.start(); // execute la méthode run()
00346         }
00347         else
00348         {
00349             communication.arreter();
00350             communication.setHandler(null);
00351             communication = null;
00352         }
00353     }

```

8.3.3.16 liberer()

```

void com.lasalle.meeting.EspaceDeTravail.liberer (
    String code )

```

Libère l'espace de travail.

Définition à la ligne 218 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.Communication.envoyer\(\)](#), [com.lasalle.meeting.Communication.fabriquerTrameModification\(\)](#), [com.lasalle.meeting.Communication.MODIFICATION_DISPONIBILITE](#), et [com.lasalle.meeting.EspaceDeTravail.setEstReserve\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherBoiteLiberation\(\)](#).

```

00219     {
00220         if(communication == null)
00221             return;
00222         Log.d(TAG, "liberer()");
00223
00224         String trame = "\0";
00225         List<String> parametres = Arrays.asList("1", code);
00226         trame = communication.fabriquerTrameModification(
Communication.MODIFICATION_DISPONIBILITE, parametres);
00227         communication.envoyer(trame, adresseIP);
00228
00229         setEstReserve(false);
00230     }

```

8.3.3.17 modifierInformations()

```

void com.lasalle.meeting.EspaceDeTravail.modifierInformations (
    List< String > parametres )

```

Modifie les informations de l'espace de travail.

Paramètres

<i>parametres</i>	champs de la trame de modification d'informations
-------------------	---

Définition à la ligne 236 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.Communication.envoyer\(\)](#), [com.lasalle.meeting.Communication.fabriquerTrameModification\(\)](#), et [com.lasalle.meeting.Communication.MODIFICATION_INFORMATIONS](#).

Référencé par [com.lasalle.meeting.ModificationEspaceDeTravail.afficherBoutons\(\)](#).

```

00237     {
00238         if (communication == null)
00239             return;
00240         Log.d(TAG, "modifierInformations()");
00241
00242         String trame = "\0";
00243         trame = communication.fabriquerTrameModification(
00244             Communication.MODIFICATION_INFORMATIONS, parametres);
00245         communication.envoyer(trame, adresseIP);
00246     }

```

8.3.3.18 reserver()

```
void com.lasalle.meeting.EspaceDeTravail.reserver ( )
```

Réserve l'espace de travail.

Définition à la ligne 201 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.Communication.envoyer\(\)](#), [com.lasalle.meeting.Communication.fabriquerTrameModification\(\)](#), [com.lasalle.meeting.Communication.MODIFICATION_DISPONIBILITE](#), et [com.lasalle.meeting.EspaceDeTravail.setEstReserve\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherBoutons\(\)](#).

```

00202     {
00203         if (communication == null)
00204             return;
00205         Log.d(TAG, "reserver()");
00206
00207         String trame = "\0";
00208         List<String> parametres = Arrays.asList("0");
00209         trame = communication.fabriquerTrameModification(
00210             Communication.MODIFICATION_DISPONIBILITE, parametres);
00211         communication.envoyer(trame, adresseIP);
00212         setEstReserve(true);
00213     }

```

8.3.3.19 setCode()

```
void com.lasalle.meeting.EspaceDeTravail.setCode (
    String code )
```

Mutateur de l'attribut code.

Paramètres

<i>code</i>	code pour libérer l'espace de travail
-------------	---------------------------------------

Définition à la ligne 178 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.code](#).

Référencé par [com.lasalle.meeting.IHMMeeting.recupererDonneesEspaceDeTravail\(\)](#).

```

00179     {
00180         this.code = code;
00181         //IHMMeeting.sauvegarderDonneesEspaceDeTravail(this);
00182
00183         Log.d(TAG, "setCode() " + code);
00184     }

```

8.3.3.20 setEstFavori()

```

void com.lasalle.meeting.EspaceDeTravail.setEstFavori (
    boolean estFavori )

```

Mutateur de l'attribut estFavori.

Paramètres

<i>estFavori</i>	si l'espace de travail est dans les favoris
------------------	---

Définition à la ligne 190 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.estFavori](#), et [com.lasalle.meeting.IHMMeeting.sauvegarderDonneesEspaceDeTravail\(\)](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherBoutons\(\)](#), et [com.lasalle.meeting.IHMMeeting.recupererDonneesEspaceDeTravail\(\)](#).

```

00191     {
00192         this.estFavori = estFavori;
00193         IHMMeeting.sauvegarderDonneesEspaceDeTravail(this);
00194
00195         Log.d(TAG, "setEstFavori() " + estFavori);
00196     }

```

8.3.3.21 setEstReserve()

```

void com.lasalle.meeting.EspaceDeTravail.setEstReserve (
    boolean estReserve )

```

Mutateur de l'attribut estReserve.

Paramètres

<i>estReserve</i>	disponibilité de l'espace de travail
-------------------	--------------------------------------

Définition à la ligne 169 du fichier [EspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.estReserve](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.liberer\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.reserver\(\)](#).

```
00170    {
00171        this.estReserve = estReserve;
00172    }
```

8.3.3.22 toJSON()

```
String com.lasalle.meeting.EspaceDeTravail.toJSON ( )
```

Création de données JSON.

Renvoie

String les données (code et favori) formaté en JSON

Définition à la ligne 367 du fichier [EspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.IHMMeeting.recupererDonneesEspaceDeTravail\(\)](#).

```
00368    {
00369        JSONObject objet = new JSONObject();
00370        try
00371        {
00372            objet.put("code", this.code);
00373            objet.put("estFavori", this.estFavori);
00374        }
00375        catch (JSONException e)
00376        {
00377            e.printStackTrace();
00378            Log.i(TAG, "toJSON() Erreur !");
00379        }
00380
00381        //Log.i(TAG, "toJSON() JSON = " + objet.toString());
00382        return objet.toString();
00383    }
```

8.3.4 Documentation des données membres

8.3.4.1 adresseIP

```
String com.lasalle.meeting.EspaceDeTravail.adresseIP [private]
```

Adresse IP du portier.

Les attributs

Définition à la ligne 44 du fichier [EspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.demanderInformations\(\)](#), [com.lasalle.meeting.EspaceDeTravail.EspaceDeTravail\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.getAdresseIP\(\)](#).

8.3.4.2 code

```
String com.lasalle.meeting.EspaceDeTravail.code [private]
```

Code pour libérer l'espace de travail.

Définition à la ligne 52 du fichier [EspaceDeTravail.java](#).

Référéncé par [com.lasalle.meeting.EspaceDeTravail.getCode\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.setCode\(\)](#).

8.3.4.3 communication

```
Communication com.lasalle.meeting.EspaceDeTravail.communication = null [private]
```

Attribut permettant d'envoyer des requêtes.

Définition à la ligne 54 du fichier [EspaceDeTravail.java](#).

8.3.4.4 description

```
String com.lasalle.meeting.EspaceDeTravail.description [private]
```

Description de l'espace de travail.

Définition à la ligne 47 du fichier [EspaceDeTravail.java](#).

Référéncé par [com.lasalle.meeting.EspaceDeTravail.getDescription\(\)](#).

8.3.4.5 estFavori

```
boolean com.lasalle.meeting.EspaceDeTravail.estFavori [private]
```

Si l'espace de travail est en favori.

Définition à la ligne 53 du fichier [EspaceDeTravail.java](#).

Référéncé par [com.lasalle.meeting.EspaceDeTravail.getEstFavori\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.setEstFavori\(\)](#).

8.3.4.6 estReserve

```
boolean com.lasalle.meeting.EspaceDeTravail.estReserve [private]
```

Disponibilité de l'espace de travail.

Définition à la ligne 51 du fichier [EspaceDeTravail.java](#).

Référéncé par [com.lasalle.meeting.EspaceDeTravail.getEstReserve\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.setEstReserve\(\)](#).

8.3.4.7 INDICE_CHAUD

```
final int com.lasalle.meeting.EspaceDeTravail.INDICE_CHAUD = 3 [static]
```

Définition à la ligne 33 du fichier [EspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherIndiceDeConfort\(\)](#), [com.lasalle.meeting.EspaceDeTravail↔Adaptateur.getView\(\)](#), [com.lasalle.meeting.IHMMeting.initialiserBoiteDeDialogueFiltrerParNiveauDeConfort\(\)](#), et [com.lasalle.↔meeting.IHMMeting.onRadioButtonClicked\(\)](#).

8.3.4.8 INDICE_FRAIS

```
final int com.lasalle.meeting.EspaceDeTravail.INDICE_FRAIS = -2 [static]
```

Définition à la ligne 38 du fichier [EspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherIndiceDeConfort\(\)](#), [com.lasalle.meeting.EspaceDeTravail↔Adaptateur.getView\(\)](#), [com.lasalle.meeting.IHMMeting.initialiserBoiteDeDialogueFiltrerParNiveauDeConfort\(\)](#), et [com.lasalle.↔meeting.IHMMeting.onRadioButtonClicked\(\)](#).

8.3.4.9 INDICE_FROID

```
final int com.lasalle.meeting.EspaceDeTravail.INDICE_FROID = -3 [static]
```

Définition à la ligne 39 du fichier [EspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherIndiceDeConfort\(\)](#), [com.lasalle.meeting.EspaceDeTravail↔Adaptateur.getView\(\)](#), [com.lasalle.meeting.IHMMeting.initialiserBoiteDeDialogueFiltrerParNiveauDeConfort\(\)](#), et [com.lasalle.↔meeting.IHMMeting.onRadioButtonClicked\(\)](#).

8.3.4.10 INDICE_LEGEREMENT_FRAIS

```
final int com.lasalle.meeting.EspaceDeTravail.INDICE_LEGEREMENT_FRAIS = -1 [static]
```

Définition à la ligne 37 du fichier [EspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherIndiceDeConfort\(\)](#), [com.lasalle.meeting.EspaceDeTravail↔Adaptateur.getView\(\)](#), [com.lasalle.meeting.IHMMeting.initialiserBoiteDeDialogueFiltrerParNiveauDeConfort\(\)](#), et [com.lasalle.↔meeting.IHMMeting.onRadioButtonClicked\(\)](#).

8.3.4.11 INDICE_LEGEREMENT_TIEDE

```
final int com.lasalle.meeting.EspaceDeTravail.INDICE_LEGEREMENT_TIEDE = 1 [static]
```

Définition à la ligne 35 du fichier [EspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.AffichageEspaceDeTravail.afficherIndiceDeConfort\(\)](#), [com.lasalle.meeting.EspaceDeTravail↔Adaptateur.getView\(\)](#), [com.lasalle.meeting.IHMMeting.initialiserBoiteDeDialogueFiltrerParNiveauDeConfort\(\)](#), et [com.lasalle.↔meeting.IHMMeting.onRadioButtonClicked\(\)](#).

8.3.4.12 INDICE_NEUTRE

```
final int com.lasalle.meeting.EspaceDeTravail.INDICE_NEUTRE = 0 [static]
```

Définition à la ligne 36 du fichier `EspaceDeTravail.java`.

Référencé par `com.lasalle.meeting.AffichageEspaceDeTravail.afficherIndiceDeConfort()`, `com.lasalle.meeting.EspaceDeTravail`↵
`Adaptateur.getView()`, `com.lasalle.meeting.IHMMeting.initialiserBoiteDeDialogueFiltrerParNiveauDeConfort()`, et `com.lasalle.`↵
`meeting.IHMMeting.onRadioButtonClicked()`.

8.3.4.13 INDICE_TIEDE

```
final int com.lasalle.meeting.EspaceDeTravail.INDICE_TIEDE = 2 [static]
```

Définition à la ligne 34 du fichier `EspaceDeTravail.java`.

Référencé par `com.lasalle.meeting.AffichageEspaceDeTravail.afficherIndiceDeConfort()`, `com.lasalle.meeting.EspaceDeTravail`↵
`Adaptateur.getView()`, `com.lasalle.meeting.IHMMeting.initialiserBoiteDeDialogueFiltrerParNiveauDeConfort()`, et `com.lasalle.`↵
`meeting.IHMMeting.onRadioButtonClicked()`.

8.3.4.14 indiceDeConfort

```
int com.lasalle.meeting.EspaceDeTravail.indiceDeConfort [private]
```

Indice de confort de l'espace de travail.

Définition à la ligne 50 du fichier `EspaceDeTravail.java`.

Référencé par `com.lasalle.meeting.EspaceDeTravail.getIndiceDeConfort()`.

8.3.4.15 lieu

```
String com.lasalle.meeting.EspaceDeTravail.lieu [private]
```

Lieu de l'espace de travail.

Définition à la ligne 46 du fichier `EspaceDeTravail.java`.

Référencé par `com.lasalle.meeting.EspaceDeTravail.getLieu()`.

8.3.4.16 nom

```
String com.lasalle.meeting.EspaceDeTravail.nom [private]
```

Nom de l'espace de travail.

Définition à la ligne 45 du fichier `EspaceDeTravail.java`.

Référencé par `com.lasalle.meeting.EspaceDeTravail.getNom()`.

8.3.4.17 superficie

```
int com.lasalle.meeting.EspaceDeTravail.superficie [private]
```

Superficie de l'espace de travail.

Définition à la ligne 48 du fichier [EspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.getSuperficie\(\)](#).

8.3.4.18 TAG

```
final String com.lasalle.meeting.EspaceDeTravail.TAG = "_EspaceDeTravail" [static], [private]
```

TAG pour les logs.

Les constantes

Définition à la ligne 32 du fichier [EspaceDeTravail.java](#).

8.3.4.19 temperature

```
double com.lasalle.meeting.EspaceDeTravail.temperature [private]
```

Température de l'espace de travail.

Définition à la ligne 49 du fichier [EspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.getTemperature\(\)](#).

La documentation de cette classe a été générée à partir du fichier suivant :

— [EspaceDeTravail.java](#)

8.4 Référence de la classe `com.lasalle.meeting.EspaceDeTravailAdaptateur`

L'affichage d'un espace de travail dans la liste des espaces de travail sur la page d'accueil.

Graphe de collaboration de `com.lasalle.meeting.EspaceDeTravailAdaptateur` :

<code>com.lasalle.meeting.EspaceDeTravailAdaptateur</code>
- TAG
+ <code>EspaceDeTravailAdaptateur()</code> + <code>getView()</code>

Classes

— class [ViewHolder](#)

Fonctions membres publiques

— [EspaceDeTravailAdaptateur](#) (Context context, int resource, Vector< [EspaceDeTravail](#) > espacesDeTravail)
— View [getView](#) (int position, View view, ViewGroup parent)

Attributs privés statiques

— static final String [TAG](#) = "_EspaceDeTravailAdaptateur"

8.4.1 Description détaillée

L'affichage d'un espace de travail dans la liste des espaces de travail sur la page d'accueil.

Définition à la ligne 27 du fichier [EspaceDeTravailAdaptateur.java](#).

8.4.2 Documentation des constructeurs et destructeur

8.4.2.1 EspaceDeTravailAdaptateur()

```
com.lasalle.meeting.EspaceDeTravailAdaptateur.EspaceDeTravailAdaptateur (
    Context context,
    int resource,
    Vector< EspaceDeTravail > espacesDeTravail )
```

Définition à la ligne 31 du fichier [EspaceDeTravailAdaptateur.java](#).

```
00032    {
00033        super(context, resource, espacesDeTravail);
00034        Log.d(TAG, "EspaceDeTravailAdaptateur()");
00035    }
```

8.4.3 Documentation des fonctions membres

8.4.3.1 getView()

```
View com.lasalle.meeting.EspaceDeTravailAdaptateur.getView (
    int position,
    View view,
    ViewGroup parent )
```

Définition à la ligne 47 du fichier [EspaceDeTravailAdaptateur.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.getDescription\(\)](#), [com.lasalle.meeting.EspaceDeTravail.getEstFavori\(\)](#), [com.lasalle.meeting.EspaceDeTravail.getEstReserve\(\)](#), [com.lasalle.meeting.EspaceDeTravail.getIndiceDeConfort\(\)](#), [com.lasalle.meeting.EspaceDeTravail.getNom\(\)](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_CHAUD](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_FRAIS](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_FROID](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_LEGEMENT_FRAIS](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_LEGEREMENT_TIEDE](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_NEUTRE](#), et [com.lasalle.meeting.EspaceDeTravail.INDICE_TIEDE](#).

```
00048     {
00049         EspaceDeTravail espaceDeTravail = null;
00050         ViewHolder viewHolder;
00051
00052         if (view == null)
00053         {
00054             viewHolder = new ViewHolder();
00055             LayoutInflater inflater = LayoutInflater.from(getContext());
00056             view = inflater.inflate(R.layout.element_espace_travail, parent, false);
00057             viewHolder.nomEspaceDeTravail = (TextView) view.findViewById(R.id.nomEspaceDeTravail);
00058             viewHolder.descriptionEspaceDeTravail = (TextView) view.findViewById(R.id.
descriptionEspaceDeTravail);
00059             viewHolder.disponibiliteEspaceDeTravail = (TextView) view.findViewById(R.id.
disponibiliteEspaceDeTravail);
00060             viewHolder.indiceDeConfortEspaceDeTravail = (TextView) view.findViewById(R.id.
indiceDeConfortEspaceDeTravail);
00061             viewHolder.favoriEspaceDeTravail = (ImageView) view.findViewById(R.id.favoriEspaceDeTravail);
00062             view.setTag(viewHolder);
00063         }
00064         else
00065         {
00066             viewHolder = (ViewHolder) view.getTag();
00067         }
00068
00069         espaceDeTravail = getItem(position);
00070         if (espaceDeTravail != null)
00071         {
00072             //Log.d(TAG, "Nom : " + espaceDeTravail.getNom());
00073             viewHolder.nomEspaceDeTravail.setText(espaceDeTravail.getNom());
00074             viewHolder.descriptionEspaceDeTravail.setText(espaceDeTravail.getDescription());
00075
00076             if(!espaceDeTravail.getEstReserve())
00077             {
00078                 viewHolder.disponibiliteEspaceDeTravail.setText("Libre");
00079                 viewHolder.disponibiliteEspaceDeTravail.setTextColor(Color.parseColor("#00FF00")); //
Color.rgb(0,255,0)
00080             }
00081             else
00082             {
00083                 viewHolder.disponibiliteEspaceDeTravail.setText("Occupé");
00084                 viewHolder.disponibiliteEspaceDeTravail.setTextColor(Color.rgb(255,0,0));
00085             }
00086
00087             switch(espaceDeTravail.getIndiceDeConfort())
00088             {
00089                 case EspaceDeTravail.INDICE_CHAUD:
00090                     viewHolder.indiceDeConfortEspaceDeTravail.setText("Chaud");
00091                     break;
00092
00093                 case EspaceDeTravail.INDICE_TIEDE:
00094                     viewHolder.indiceDeConfortEspaceDeTravail.setText("Tiède");
00095                     break;
00096
00097                 case EspaceDeTravail.INDICE_LEGEREMENT_TIEDE:
00098                     viewHolder.indiceDeConfortEspaceDeTravail.setText("Légèrement tiède");
00099                     break;
00100
00101                 case EspaceDeTravail.INDICE_NEUTRE:
00102                     viewHolder.indiceDeConfortEspaceDeTravail.setText("Neutre");
00103                     break;
00104
00105                 case EspaceDeTravail.INDICE_LEGEREMENT_FRAIS:
00106                     viewHolder.indiceDeConfortEspaceDeTravail.setText("Légèrement frais");
00107                     break;
00108
00109                 case EspaceDeTravail.INDICE_FRAIS:
```

```
00110         viewHolder.indiceDeConfortEspaceDeTravail.setText("Frais");
00111         break;
00112
00113         case EspaceDeTravail.INDICE_FROID:
00114             viewHolder.indiceDeConfortEspaceDeTravail.setText("Froid");
00115             break;
00116     }
00117
00118     if(!espaceDeTravail.getEstFavori())
00119     {
00120         viewHolder.favoriEspaceDeTravail.setVisibility(View.INVISIBLE);
00121     }
00122     else
00123     {
00124         viewHolder.favoriEspaceDeTravail.setVisibility(View.VISIBLE);
00125     }
00126 }
00127
00128 return view;
00129 }
```

8.4.4 Documentation des données membres

8.4.4.1 TAG

```
final String com.lasalle.meeting.EspaceDeTravailAdaptateur.TAG = "_EspaceDeTravailAdaptateur" [static], [private]
```

Définition à la ligne 29 du fichier [EspaceDeTravailAdaptateur.java](#).

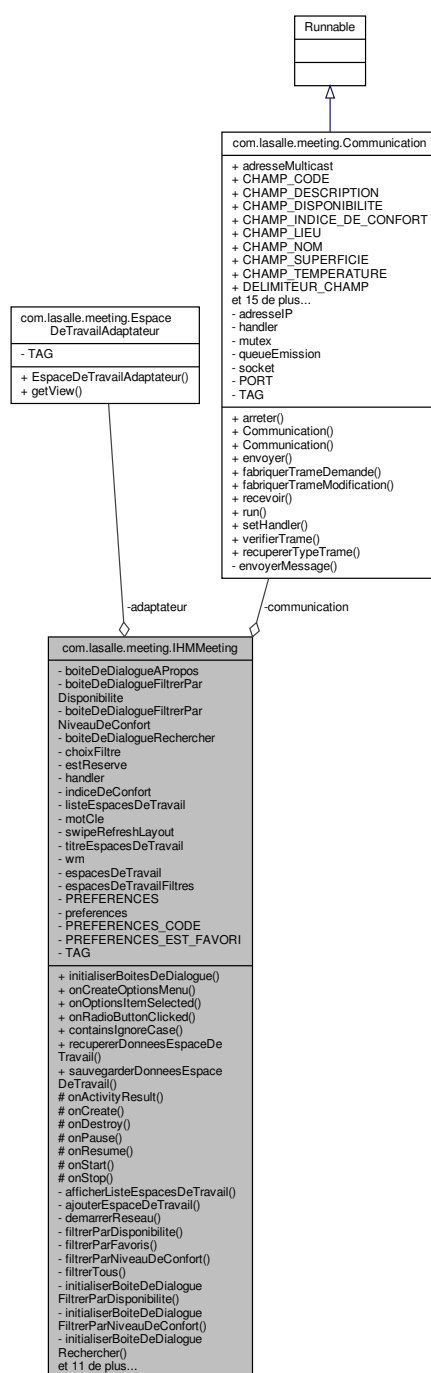
La documentation de cette classe a été générée à partir du fichier suivant :

— [EspaceDeTravailAdaptateur.java](#)

8.5 Référence de la classe com.lasalle.meeting.IHMMeeting

L'activité principale de l'application Meeting.

Graphe de collaboration de com.lasalle.meeting.IHMMeeting :



Fonctions membres publiques

- void `initialiserBoitesDeDialogue()`
Initialise les boites de dialogue.
- boolean `onCreateOptionsMenu` (Menu menu)
Méthode pour la création du menu.
- boolean `onOptionsItemSelected` (MenuItem item)
Méthode appelée lors de la sélection d'une entrée de menu.
- void `onRadioButtonClicked` (View vue)
Méthode appelée lorsque l'on coche un bouton radio.

Fonctions membres publiques statiques

- static boolean [containsIgnoreCase](#) (String str, String searchStr)
Utilitaire de recherche insensible à la casse.
- static String [recupererDonneesEspaceDeTravail](#) ([EspaceDeTravail](#) espaceDeTravail)
Récupère les données stockées d'un espace de travail.
- static void [sauvegarderDonneesEspaceDeTravail](#) ([EspaceDeTravail](#) espaceDeTravail)
Enregistre les données d'un espace de travail.

Fonctions membres protégées

- void [onActivityResult](#) (int requestCode, int resultCode, Intent data)
Traite le retour de l'activité d'affichage d'un espace de travail.
- void [onCreate](#) (Bundle savedInstanceState)
Méthode appelée à la création de l'activité
- void [onDestroy](#) ()
Méthode appelée à la destruction de l'application (après [onStop\(\)](#) et détruite par le système Android)
- void [onPause](#) ()
Méthode appelée après qu'une boîte de dialogue s'est affichée (on reprend sur un [onResume\(\)](#)) ou avant [onStop\(\)](#) (activité plus visible)
- void [onResume](#) ()
Méthode appelée après [onStart\(\)](#) ou après [onPause\(\)](#)
- void [onStart](#) ()
Méthode appelée au démarrage de l'activité MainActivity.
- void [onStop](#) ()
Méthode appelée lorsque l'activité n'est plus visible.

Fonctions membres privées

- void [afficherListeEspacesDeTravail](#) ()
Affiche la liste des espaces de travail.
- void [ajouterEspaceDeTravail](#) ([EspaceDeTravail](#) espaceDeTravail)
Ajoute un espace de travail si pas encore détecté
- void [demarrerReseau](#) ()
Démarré la connexion wifi et la communication.
- void [filtrerParDisponibilite](#) (boolean disponibilite)
Filtre les espaces de travail par disponibilité
- void [filtrerParFavoris](#) ()
Filtre les espaces de travail par favori.
- void [filtrerParNiveauDeConfort](#) (int indiceDeConfort)
Filtre les espaces de travail par niveau de confort.
- void [filtrerTous](#) ()
Récupère tous les espaces de travail détectés.
- void [initialiserBoiteDeDialogueFiltrerParDisponibilite](#) ()
Initialise la boîte de dialogue de filtrage par disponibilité
- void [initialiserBoiteDeDialogueFiltrerParNiveauDeConfort](#) ()
Initialise la boîte de dialogue de filtrage par niveau de confort.
- void [initialiserBoiteDeDialogueRechercher](#) ()
Initialise la boîte de dialogue de recherche.
- void [initialiserEspacesDeTravail](#) ()
Crée les espaces de travail détectés.
- void [initialiserListeEspacesDeTravail](#) ()
Initialise la vue pour les espaces de travail.
- void [initialiserPreferences](#) ()
Initialise les préférences.
- void [modifierEspaceDeTravail](#) ([EspaceDeTravail](#) espaceDeTravail)
Modifie un espace de travail si déjà détecté
- void [rafraichirAffichageListeEspaces](#) ()
Met à jour l'affichage de la liste.
- void [rafraichirListeEspacesFiltres](#) ()
Fabrique la liste des espaces de travail à afficher.
- void [rechercher](#) (String motCle)
Recherche les espaces de travail par un mot-clé
- void [supprimerEspaceDeTravail](#) ([EspaceDeTravail](#) espaceDeTravail)
Supprime un espace de travail si déjà détecté
- void [trierEspacesDeTravail](#) (final String champ, Vector< [EspaceDeTravail](#) > lesEspacesDeTravail)
Trie les espaces de travail.
- int [verifierPresenceEspaceDeTravail](#) ([EspaceDeTravail](#) espaceDeTravail)
Vérifie la présence d'un espace de travail dans le conteneur des espaces de travail détectés.
- int [verifierPresenceEspaceDeTravail](#) (String adresseIP)
Vérifie la présence d'un espace de travail dans le conteneur des espaces de travail détectés.

Attributs privés

- [EspaceDeTravailAdaptateur](#) adaptateur
Adaptateur pour les espaces de travail.
- AlertDialog.Builder [boiteDeDialogueAPropos](#)
Boîte de dialogue A propos.
- AlertDialog.Builder [boiteDeDialogueFiltrerParDisponibilite](#)
Boîte de dialogue Filtrer par disponibilité
- AlertDialog.Builder [boiteDeDialogueFiltrerParNiveauDeConfort](#)
Boîte de dialogue Filtrer par niveau de confort.
- AlertDialog [boiteDeDialogueRechercher](#)
Boîte de dialogue Rechercher.
- int [choixFiltre](#) = -1
Choix du dernier type de filtrage.
- boolean [estReserve](#)
- Handler [handler](#)
Permet de récupérer les trames.
- int [indiceDeConfort](#)
- ListView [listeEspacesDeTravail](#)
Affichage des espaces de travail sous forme de liste.
- String [motCle](#)
Mot-clé pour rechercher un espace de travail.
- SwipeRefreshLayout [swipeRefreshLayout](#)
Pour le Pull To Refresh.
- TextView [titreEspacesDeTravail](#)
Affichage du titre principal.
- WifiManager [wm](#) = null
Attribut permettant de voir la connexion au WiFi.

Attributs privés statiques

- static [Communication](#) [communication](#) = null
Attribut permettant d'envoyer des requêtes.
- static Vector< [EspaceDeTravail](#) > [espacesDeTravail](#)
Conteneur pour les espaces de travail.
- static Vector< [EspaceDeTravail](#) > [espacesDeTravailFiltres](#)
Conteneur pour les espaces de travail filtrés.
- static final String [PREFERENCES](#) = "preferences"
Préférences de l'application.
- static SharedPreferences [preferences](#)
Pour le stockage de données.
- static final String [PREFERENCES_CODE](#) = "code"
Code pour libérer l'espace de travail enregistré dans les préférences.
- static final String [PREFERENCES_EST_FAVORI](#) = "false"
Favori enregistré dans les préférences.
- static final String [TAG](#) = "_IHMMeting"
TAG pour les logs.

8.5.1 Description détaillée

L'activité principale de l'application Meeting.

Définition à la ligne 44 du fichier [IHMMeting.java](#).

8.5.2 Documentation des fonctions membres

8.5.2.1 afficherListeEspacesDeTravail()

```
void com.lasalle.meeting.IHMMeeting.afficherListeEspacesDeTravail ( ) [private]
```

Affiche la liste des espaces de travail.

Définition à la ligne 319 du fichier [IHMMeeting.java](#).

Référencé par [com.lasalle.meeting.IHMMeeting.onCreate\(\)](#).

```
00320     {
00321         Log.d(TAG, "afficherListeEspacesDeTravail()");
00322
00323         adaptateur = new EspaceDeTravailAdaptateur(this, R.layout.element_espace_travail,
00324             espacesDeTravailFiltres);
00325         listeEspacesDeTravail.setAdapter(adaptateur);
00326         adaptateur.setNotifyOnChange(true);
00327
00328         listeEspacesDeTravail.setOnItemClickListener(
00329             new AdapterView.OnItemClickListener()
00330             {
00331                 @Override
00332                 public void onItemClick(AdapterView<?> a, View v, int position, long id)
00333                 {
00334                     Log.d(TAG, "Position : " + position + " - " + " Nom : " +
00335                         espacesDeTravailFiltres.get(position).getNom());
00336                     Intent intent = new Intent(IHMMeeting.this, AffichageEspaceDeTravail.class);
00337                     intent.putExtra("unEspaceDeTravail", (Serializable)
00338                         espacesDeTravailFiltres.get(position));
00339                     startActivityForResult(intent, 0);
00340                 }
00341             });
00342     }
```

8.5.2.2 ajouterEspaceDeTravail()

```
void com.lasalle.meeting.IHMMeeting.ajouterEspaceDeTravail (
    EspaceDeTravail espaceDeTravail ) [private]
```

Ajoute un espace de travail si pas encore détecté

Paramètres

<i>espaceDeTravail</i>	un espace de travail
------------------------	----------------------

Définition à la ligne 222 du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.IHMMeeting.rafraichirListeEspacesFiltres\(\)](#), et [com.lasalle.meeting.IHMMeeting.verifierPresenceEspaceDeTravail\(\)](#).

```
00223     {
00224         int position = verifierPresenceEspaceDeTravail(espaceDeTravail);
00225
00226         if(position == -1)
00227         {
00228             espacesDeTravail.add(espaceDeTravail);
00229             rafraichirListeEspacesFiltres();
00230         }
00231     }
```

8.5.2.3 containsIgnoreCase()

```
static boolean com.lasalle.meeting.IHMMeeting.containsIgnoreCase (
    String str,
    String searchStr ) [static]
```

Utilitaire de recherche insensible à la casse.

Auteur

Thierry Vaira

Définition à la ligne 884 du fichier [IHMMeeting.java](#).

Référencé par [com.lasalle.meeting.IHMMeeting.rechercher\(\)](#).

```
00885     {
00886         if(str == null || searchStr == null) return false;
00887
00888         final int length = searchStr.length();
00889         if (length == 0)
00890             return true;
00891
00892         for (int i = str.length() - length; i >= 0; i--)
00893         {
00894             if (str.regionMatches(true, i, searchStr, 0, length))
00895                 return true;
00896         }
00897         return false;
00898     }
```

8.5.2.4 demarrerReseau()

```
void com.lasalle.meeting.IHMMeeting.demarrerReseau ( ) [private]
```

Démarre la connexion wifi et la communication.

Définition à la ligne 122 du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.Communication.adresseMulticast](#), [com.lasalle.meeting.Communication.DEMANDE_INFORMATIO↵](#)
[NS](#), [com.lasalle.meeting.Communication.envoyer\(\)](#), [com.lasalle.meeting.Communication.fabriquerTrameDemande\(\)](#), et [com.lasalle.↵](#)
[meeting.IHMMeeting.handler](#).

Référencé par [com.lasalle.meeting.IHMMeeting.onCreate\(\)](#).

```
00123     {
00124         wm = (WifiManager) getApplicationContext().getSystemService(Context.WIFI_SERVICE);
00125         if (!wm.isWifiEnabled())
00126         {
00127             Log.d(TAG, "WiFi indisponible !");
00128             wm.setWifiEnabled(true);
00129         }
00130         else
00131         {
00132             Log.d(TAG, "WiFi disponible");
00133         }
00134
00135         communication = new Communication(handler);
00136
00137         // Démarre la réception des trames des portiers
00138         Thread tCommunicationUDP = new Thread(communication, "Communication");
00139         tCommunicationUDP.start(); // execute la méthode run()
00140
00141         if(communication != null)
00142         {
00143             // Demande les informations des portiers joignables
00144             communication.envoyer(communication.fabriquerTrameDemande(Communication.DEMANDE_INFORMATIONS), Communication.adresseMulticast);
00145         }
00146     }
```

8.5.2.5 filtrerParDisponibilite()

```
void com.lasalle.meeting.IHMMeeting.filtrerParDisponibilite (
    boolean disponibilite ) [private]
```

Filtre les espaces de travail par disponibilité

Définition à la ligne 714 du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.IHMMeeting.rafraichirAffichageListeEspaces\(\)](#).

Référencé par [com.lasalle.meeting.IHMMeeting.initialiserBoiteDeDialogueFiltrerParDisponibilite\(\)](#), et [com.lasalle.meeting.IHMMeeting.rafraichirListeEspacesFiltres\(\)](#).

```
00715     {
00716         espacesDeTravailFiltres.clear();
00717
00718         for(int i = 0; i < espacesDeTravail.size(); i++)
00719         {
00720             if(espacesDeTravail.elementAt(i).getEstReserve() == disponibilite)
00721             {
00722                 espacesDeTravailFiltres.add(
00723                     espacesDeTravail.elementAt(i));
00724             }
00725
00726             Log.d(TAG, "filtrerParDisponibilite() " + disponibilite);
00727
00728             rafraichirAffichageListeEspaces();
00729     }
```

8.5.2.6 filtrerParFavoris()

```
void com.lasalle.meeting.IHMMeeting.filtrerParFavoris ( ) [private]
```

Filtre les espaces de travail par favori.

Définition à la ligne 754 du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.IHMMeeting.rafraichirAffichageListeEspaces\(\)](#).

Référencé par [com.lasalle.meeting.IHMMeeting.onOptionsItemSelected\(\)](#), et [com.lasalle.meeting.IHMMeeting.rafraichirListeEspacesFiltres\(\)](#).

```
00755     {
00756         espacesDeTravailFiltres.clear();
00757
00758         for(int i = 0; i < espacesDeTravail.size(); i++)
00759         {
00760             if(espacesDeTravail.elementAt(i).getEstFavori())
00761             {
00762                 espacesDeTravailFiltres.add(
00763                     espacesDeTravail.elementAt(i));
00764             }
00765
00766             Log.d(TAG, "filtrerParFavoris()");
00767
00768             rafraichirAffichageListeEspaces();
00769     }
```

8.5.2.7 filtrerParNiveauDeConfort()

```
void com.lasalle.meeting.IHMMeeting.filtrerParNiveauDeConfort (
    int indiceDeConfort ) [private]
```

Filtre les espaces de travail par niveau de confort.

Définition à la ligne 734 du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.IHMMeeting.indiceDeConfort](#), et [com.lasalle.meeting.IHMMeeting.rafraichirAffichageListeEspaces\(\)](#).

Référencé par [com.lasalle.meeting.IHMMeeting.initialiserBoiteDeDialogueFiltrerParNiveauDeConfort\(\)](#), et [com.lasalle.meeting.IHMMeeting.rafraichirListeEspacesFiltres\(\)](#).

```
00735     {
00736         espacesDeTravailFiltres.clear();
00737
00738         for(int i = 0; i < espacesDeTravail.size(); i++)
00739         {
00740             if(espacesDeTravail.elementAt(i).getIndiceDeConfort() ==
00741                indiceDeConfort)
00742             {
00743                 espacesDeTravailFiltres.add(
00744                     espacesDeTravail.elementAt(i));
00745             }
00746             Log.d(TAG, "filtrerParNiveauDeConfort() " + indiceDeConfort);
00747             rafraichirAffichageListeEspaces();
00748         }
00749     }
```

8.5.2.8 filtrerTous()

```
void com.lasalle.meeting.IHMMeeting.filtrerTous ( ) [private]
```

Récupère tous les espaces de travail détectés.

Définition à la ligne 774 du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.IHMMeeting.rafraichirAffichageListeEspaces\(\)](#).

Référencé par [com.lasalle.meeting.IHMMeeting.onOptionsItemSelected\(\)](#), et [com.lasalle.meeting.IHMMeeting.rafraichirListeEspacesFiltres\(\)](#).

```
00775     {
00776         espacesDeTravailFiltres.clear();
00777
00778         for(int i = 0; i < espacesDeTravail.size(); i++)
00779         {
00780             espacesDeTravailFiltres.add(espacesDeTravail.elementAt(i));
00781         }
00782
00783         Log.d(TAG, "filtrerTous");
00784         rafraichirAffichageListeEspaces();
00785     }
00786 }
```

8.5.2.9 initialiserBoiteDeDialogueFiltrerParDisponibilite()

```
void com.lasalle.meeting.IHMMeeting.initialiserBoiteDeDialogueFiltrerParDisponibilite ( ) [private]
```

Initialise la boite de dialogue de filtrage par disponibilité

Définition à la ligne 526 du fichier IHMMeeting.java.

Références [com.lasalle.meeting.IHMMeeting.boiteDeDialogueFiltrerParDisponibilite](#), et [com.lasalle.meeting.IHMMeeting.filtrerParDisponibilite\(\)](#).

Référencé par [com.lasalle.meeting.IHMMeeting.initialiserBoitesDeDialogue\(\)](#).

```
00527     {
00528         boiteDeDialogueFiltrerParDisponibilite = new AlertDialog.
Builder(this);
00529         boiteDeDialogueFiltrerParDisponibilite.setTitle("Filtrer par
disponibilité");
00530         LayoutInflater inflater = this.getLayoutInflater();
00531         View vue = inflater.inflate(R.layout.boite_filtre_disponibilite, null);
00532         boiteDeDialogueFiltrerParDisponibilite.setView(vue);
00533
00534         boiteDeDialogueFiltrerParDisponibilite.setPositiveButton("
Filtrer", new DialogInterface.OnClickListener()
00535         {
00536             public void onClick(DialogInterface dialog, int which)
00537             {
00538                 if(estReserve)
00539                 {
00540                     titreEspacesDeTravail.setText("Espaces de travail occupés");
00541                 }
00542                 else
00543                 {
00544                     titreEspacesDeTravail.setText("Espaces de travail libres");
00545                 }
00546
00547                 filtrerParDisponibilite(estReserve);
00548             }
00549         });
00550         boiteDeDialogueFiltrerParDisponibilite.setNegativeButton("
Annuler", new DialogInterface.OnClickListener()
00551         {
00552             public void onClick(DialogInterface dialog, int which)
00553             {
00554
00555             }
00556         });
00557     }
```

8.5.2.10 initialiserBoiteDeDialogueFiltrerParNiveauDeConfort()

```
void com.lasalle.meeting.IHMMeeting.initialiserBoiteDeDialogueFiltrerParNiveauDeConfort ( ) [private]
```

Initialise la boite de dialogue de filtrage par niveau de confort.

Définition à la ligne 562 du fichier IHMMeeting.java.

Références [com.lasalle.meeting.IHMMeeting.boiteDeDialogueFiltrerParNiveauDeConfort](#), [com.lasalle.meeting.IHMMeeting.filtrerParNiveauDeConfort\(\)](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_CHAUD](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_FRAIS](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_FROID](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_LEGEREMENT_FRAIS](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_LEGEREMENT_TIEDE](#), [com.lasalle.meeting.EspaceDeTravail.INDICE_NEUTRE](#), et [com.lasalle.meeting.EspaceDeTravail.INDICE_TIEDE](#).

Référencé par [com.lasalle.meeting.IHMMeeting.initialiserBoitesDeDialogue\(\)](#).

```

00563     {
00564         boiteDeDialogueFiltrerParNiveauDeConfort = new AlertDialog.
Builder(this);
00565         boiteDeDialogueFiltrerParNiveauDeConfort.setTitle("Filtrer
par niveau de confort");
00566         LayoutInflater inflater = this.getLayoutInflater();
00567         View vue = inflater.inflate(R.layout.boite_filtre_niveau_de_confort, null);
00568         boiteDeDialogueFiltrerParNiveauDeConfort.setView(vue);
00569
00570         boiteDeDialogueFiltrerParNiveauDeConfort.setPositiveButton(
"Filtrer", new DialogInterface.OnClickListener()
00571         {
00572             public void onClick(DialogInterface dialog, int which)
00573             {
00574                 switch(indiceDeConfort)
00575                 {
00576                     case EspaceDeTravail.INDICE_CHAUD:
00577                         titreEspacesDeTravail.setText("Espaces de travail dont le
niveau de confort est Chaud");
00578                         break;
00579
00580                     case EspaceDeTravail.INDICE_TIEDE:
00581                         titreEspacesDeTravail.setText("Espaces de travail dont le
niveau de confort est Tiède");
00582                         break;
00583
00584                     case EspaceDeTravail.INDICE_LEGEREMENT_TIEDE:
00585                         titreEspacesDeTravail.setText("Espaces de travail dont le
niveau de confort est Légèrement tiède");
00586                         break;
00587
00588                     case EspaceDeTravail.INDICE_NEUTRE:
00589                         titreEspacesDeTravail.setText("Espaces de travail dont le
niveau de confort est Neutre");
00590                         break;
00591
00592                     case EspaceDeTravail.INDICE_LEGEREMENT_FRAIS:
00593                         titreEspacesDeTravail.setText("Espaces de travail dont le
niveau de confort est Légèrement frais");
00594                         break;
00595
00596                     case EspaceDeTravail.INDICE_FRAIS:
00597                         titreEspacesDeTravail.setText("Espaces de travail dont le
niveau de confort est Frais");
00598                         break;
00599
00600                     case EspaceDeTravail.INDICE_FROID:
00601                         titreEspacesDeTravail.setText("Espaces de travail dont le
niveau de confort est Froid");
00602                         break;
00603                 }
00604             }
00605             filtrerParNiveauDeConfort(
indiceDeConfort);
00606         }
00607     });
00608     boiteDeDialogueFiltrerParNiveauDeConfort.setNegativeButton(
"Annuler", new DialogInterface.OnClickListener()
00609     {
00610         public void onClick(DialogInterface dialog, int which)
00611         {
00612
00613         }
00614     });
00615 }

```

8.5.2.11 initialiserBoiteDeDialogueRechercher()

```
void com.lasalle.meeting.IHMMeeting.initialiserBoiteDeDialogueRechercher ( ) [private]
```

Initialise la boîte de dialogue de recherche.

Définition à la ligne 494 du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.IHMMeeting.rechercher\(\)](#).

Référencé par [com.lasalle.meeting.IHMMeeting.initialiserBoitesDeDialogue\(\)](#).

```

00495     {
00496         AlertDialog.Builder boiteDeDialogueRechercher = new AlertDialog.Builder(
00497             this);
00498         boiteDeDialogueRechercher.setTitle("Rechercher un espace de travail");
00499         LayoutInflater inflater = this.getLayoutInflater();
00500         View vue = inflater.inflate(R.layout.boite_recherche, null);
00501         boiteDeDialogueRechercher.setView(vue);
00502         boiteDeDialogueRechercher.setPositiveButton("Rechercher", new
DialogInterface.OnClickListener()
00503         {
00504             public void onClick(DialogInterface dialog, int which)
00505             {
00506                 EditText saisieMotCle = (EditText) ((AlertDialog) dialog).findViewById(R.id.saisieMotCle);
00507                 motCle = saisieMotCle.getText().toString();
00508                 titreEspacesDeTravail.setText("Résultats de la recherche : " +
motCle);
00509                 rechercher(motCle);
00510             }
00511         });
00512         boiteDeDialogueRechercher.setNegativeButton("Annuler", new DialogInterface
.OnClickListener()
00513         {
00514             public void onClick(DialogInterface dialog, int which)
00515             {
00516             }
00517         });
00518     });
00519     this.boiteDeDialogueRechercher =
boiteDeDialogueRechercher.create();
00521 }

```

8.5.2.12 initialiserBoitesDeDialogue()

```
void com.lasalle.meeting.IHMMeeting.initialiserBoitesDeDialogue ( )
```

Initialise les boites de dialogue.

Définition à la ligne 479 du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.IHMMeeting.boiteDeDialogueAPropos](#), [com.lasalle.meeting.IHMMeeting.initialiserBoiteDeDialogueFiltrerParDisponibilite\(\)](#), [com.lasalle.meeting.IHMMeeting.initialiserBoiteDeDialogueFiltrerParNiveauDeConfort\(\)](#), et [com.lasalle.meeting.IHMMeeting.initialiserBoiteDeDialogueRechercher\(\)](#).

Référencé par [com.lasalle.meeting.IHMMeeting.onCreate\(\)](#).

```

00480     {
00481         boiteDeDialogueAPropos = new AlertDialog.Builder(this);
00482         boiteDeDialogueAPropos.setTitle("À propos");
00483         String message = "Projet Meeting version " + BuildConfig.VERSION_NAME + "\n\nAuteur :
KELLER-LAVALLEE Joachim\nBTS SNIR La Salle Avignon 2021";
00484         boiteDeDialogueAPropos.setMessage(message);
00485
00486         initialiserBoiteDeDialogueRechercher();
00487         initialiserBoiteDeDialogueFiltrerParDisponibilite(
00488         );
00489         initialiserBoiteDeDialogueFiltrerParNiveauDeConfort
00490         ();
00491     }

```

8.5.2.13 initialiserEspacesDeTravail()

```
void com.lasalle.meeting.IHMMeeting.initialiserEspacesDeTravail ( ) [private]
```

Crée les espaces de travail détectés.

Définition à la ligne 203 du fichier `IHMMeeting.java`.

Références `com.lasalle.meeting.Communication.adresseMulticast`, `com.lasalle.meeting.Communication.DEMANDE_INFORMATIO←NS`, `com.lasalle.meeting.Communication.envoyer()`, et `com.lasalle.meeting.Communication.fabriquerTrameDemande()`.

Référencé par `com.lasalle.meeting.IHMMeeting.onCreate()`, et `com.lasalle.meeting.IHMMeeting.onStart()`.

```
00204     {
00205         Log.d(TAG, "initialiserEspacesDeTravail()");
00206
00207         espacesDeTravail.clear();
00208
00209         if (communication != null)
00210         {
00211             // Demande les informations des portiers joignables
00212             communication.envoyer (communication.
fabriquerTrameDemande (Communication.DEMANDE_INFORMATIONS), Communication.
adresseMulticast);
00213         }
00214
00215         swipeRefreshLayout.setRefreshing(false); // arrête le Pull To Refresh
00216     }
```

8.5.2.14 initialiserListeEspacesDeTravail()

```
void com.lasalle.meeting.IHMMeeting.initialiserListeEspacesDeTravail ( ) [private]
```

Initialise la vue pour les espaces de travail.

Définition à la ligne 303 du fichier `IHMMeeting.java`.

Référencé par `com.lasalle.meeting.IHMMeeting.onCreate()`.

```
00304     {
00305         Log.d(TAG, "initialiserListeEspacesDeTravail()");
00306
00307         espacesDeTravail = new Vector<EspaceDeTravail>();
00308         espacesDeTravailFiltres = new Vector<EspaceDeTravail>();
00309         choixFiltre = R.id.actionAfficherTous;
00310
00311         listeEspacesDeTravail = (ListView) findViewById(R.id.listeEspacesDeTravail);
00312         titreEspacesDeTravail = (TextView) findViewById(R.id.titreEspacesDeTravail);
00313         titreEspacesDeTravail.setText("Tous les espaces de travail détectés");
00314     }
```

8.5.2.15 initialiserPreferences()

```
void com.lasalle.meeting.IHMMeeting.initialiserPreferences ( ) [private]
```

Initialise les préférences.

Définition à la ligne 114 du fichier `IHMMeeting.java`.

Référencé par `com.lasalle.meeting.IHMMeeting.onCreate()`.

```
00115     {
00116         preferences = getBaseContext().getSharedPreferences(
PREFERENCES, MODE_PRIVATE);
00117     }
```

8.5.2.16 modifierEspaceDeTravail()

```
void com.lasalle.meeting.IHMMeeting.modifierEspaceDeTravail (
    EspaceDeTravail espaceDeTravail ) [private]
```

Modifie un espace de travail si déjà détecté

Paramètres

<code>espaceDeTravail</code>	l'espace de travail modifié
------------------------------	-----------------------------

Définition à la ligne 237 du fichier `IHMMeeting.java`.

Références `com.lasalle.meeting.IHMMeeting.rafraichirListeEspacesFiltres()`, et `com.lasalle.meeting.IHMMeeting.verifierPresenceEspaceDeTravail()`.

```

00238    {
00239        int position = verifierPresenceEspaceDeTravail(espaceDeTravail);
00240        Log.d(TAG, "modifierEspaceDeTravail() : position = " + position);
00241
00242        if(position != -1)
00243        {
00244            //espacesDeTravail.removeElementAt(position);
00245            //espacesDeTravail.add(espaceDeTravail);
00246            espacesDeTravail.set(position, espaceDeTravail);
00247            rafraichirListeEspacesFiltres();
00248        }
00249    }

```

8.5.2.17 `onActivityResult()`

```

void com.lasalle.meeting.IHMMeeting.onActivityResult (
    int requestCode,
    int resultCode,
    Intent data ) [protected]

```

Traite le retour de l'activité d'affichage d'un espace de travail.

Définition à la ligne 347 du fichier `IHMMeeting.java`.

```

00348    {
00349        super.onActivityResult(requestCode, resultCode, data);
00350        Log.d(TAG, "onActivityResult() requestCode=" + requestCode + " - resultCode=" + resultCode + "");
00351        ;
00352        /*EspaceDeTravail espaceDeTravail =
00353        (EspaceDeTravail)data.getSerializableExtra("unEspaceDeTravail");
00354        Log.d(TAG, "onActivityResult() espaceDeTravail : " + espaceDeTravail.getNom() + " - " +
00355        espaceDeTravail.getDescription() + " - " + espaceDeTravail.getLieu() + " - " + espaceDeTravail.getSuperficie() + " - "
00356        + espaceDeTravail.getEstReserve());
00357        modifierEspaceDeTravail(espaceDeTravail);*/
00358    }

```

8.5.2.18 `onCreate()`

```

void com.lasalle.meeting.IHMMeeting.onCreate (
    Bundle savedInstanceState ) [protected]

```

Méthode appelée à la création de l'activité

Définition à la ligne 83 du fichier `IHMMeeting.java`.

Références `com.lasalle.meeting.IHMMeeting.afficherListeEspacesDeTravail()`, `com.lasalle.meeting.IHMMeeting.demarrerReseau()`, `com.lasalle.meeting.IHMMeeting.initialiserBoitesDeDialogue()`, `com.lasalle.meeting.IHMMeeting.initialiserEspacesDeTravail()`, `com.lasalle.meeting.IHMMeeting.initialiserListeEspacesDeTravail()`, et `com.lasalle.meeting.IHMMeeting.initialiserPreferences()`.

```

00084    {
00085        super.onCreate(savedInstanceState);
00086        setContentView(R.layout.activity_main);
00087        Log.d(TAG, "onCreate()");
00088
00089        initialiserPreferences();
00090
00091        swipeRefreshLayout = (SwipeRefreshLayout) findViewById(R.id.swipeRefreshLayout);
00092        swipeRefreshLayout.setOnRefreshListener(new SwipeRefreshLayout.OnRefreshListener()
00093    )
00094    {
00095        @Override
00096        public void onRefresh()
00097        {
00098            Log.d(TAG, "Pull To Refresh");
00099            initialiserEspacesDeTravail();
00100        }
00101    });
00102
00103    initialiserListeEspacesDeTravail();
00104
00105    afficherListeEspacesDeTravail();
00106
00107    demarrerReseau();
00108
00109    initialiserBoitesDeDialogue();
00110 }

```

8.5.2.19 onCreateOptionsMenu()

```

boolean com.lasalle.meeting.IHMMeeting.onCreateOptionsMenu (
    Menu menu )

```

Méthode pour la création du menu.

Définition à la ligne 414 du fichier [IHMMeeting.java](#).

```

00415    {
00416        getMenuInflater().inflate(R.menu.menu_main, menu);
00417        return true;
00418    }

```

8.5.2.20 onDestroy()

```

void com.lasalle.meeting.IHMMeeting.onDestroy ( ) [protected]

```

Méthode appelée à la destruction de l'application (après [onStop\(\)](#) et détruite par le système Android)

Définition à la ligne 194 du fichier [IHMMeeting.java](#).

```

00195    {
00196        super.onDestroy();
00197        Log.d(TAG, "onDestroy()");
00198    }

```

8.5.2.21 onOptionsItemSelected()

```
boolean com.lasalle.meeting.IHMMeeting.onOptionsItemSelected (
    MenuItem item )
```

Méthode appelée lors de la sélection d'une entrée de menu.

Définition à la ligne 424 du fichier `IHMMeeting.java`.

Références `com.lasalle.meeting.IHMMeeting.boiteDeDialogueAPropos`, `com.lasalle.meeting.IHMMeeting.boiteDeDialogueFiltrer`↵
`ParDisponibilite`, `com.lasalle.meeting.IHMMeeting.boiteDeDialogueFiltrerParNiveauDeConfort`, `com.lasalle.meeting.IHMMeeting`↵
`boiteDeDialogueRechercher`, `com.lasalle.meeting.IHMMeeting.filtrerParFavoris()`, et `com.lasalle.meeting.IHMMeeting.filtrerTous()`.

```
00425     {
00426         int id = item.getItemId();
00427
00428         switch(id)
00429         {
00430             case R.id.actionAfficherTous:
00431                 Log.d(TAG, "onOptionsItemSelected() actionAfficherTous");
00432                 choixFiltre = R.id.actionAfficherTous;
00433                 titreEspacesDeTravail.setText("Tous les espaces de travail détectés");
00434                 filtrerTous();
00435                 break;
00436
00437             case R.id.actionAfficherFavoris:
00438                 Log.d(TAG, "onOptionsItemSelected() actionAfficherFavoris");
00439                 choixFiltre = R.id.actionAfficherFavoris;
00440                 titreEspacesDeTravail.setText("Favoris");
00441                 filtrerParFavoris();
00442                 break;
00443
00444             case R.id.actionRechercher:
00445                 Log.d(TAG, "onOptionsItemSelected() actionRechercher");
00446                 choixFiltre = R.id.actionRechercher;
00447                 titreEspacesDeTravail.setText("Recherche");
00448                 boiteDeDialogueRechercher.show();
00449                 EditText saisieMotCle = (EditText) ((AlertDialog)
boiteDeDialogueRechercher).findViewById(R.id.saisieMotCle);
00450                 saisieMotCle.setText(motCle);
00451                 break;
00452
00453             case R.id.actionFiltrerParDisponibilite:
00454                 Log.d(TAG, "onOptionsItemSelected() actionFiltrerParDisponibilite");
00455                 choixFiltre = R.id.actionFiltrerParDisponibilite;
00456                 titreEspacesDeTravail.setText("Espaces de travail filtrés par
disponibilité");
00457                 boiteDeDialogueFiltrerParDisponibilite.show();
00458                 break;
00459
00460             case R.id.actionFiltrerParNiveauDeConfort:
00461                 Log.d(TAG, "onOptionsItemSelected() actionFiltrerParNiveauDeConfort");
00462                 choixFiltre = R.id.actionFiltrerParNiveauDeConfort;
00463                 titreEspacesDeTravail.setText("Espaces de travail filtrés par niveau
de confort");
00464                 boiteDeDialogueFiltrerParNiveauDeConfort.show();
00465                 break;
00466
00467             case R.id.actionAPropos:
00468                 Log.d(TAG, "onOptionsItemSelected() actionAPropos");
00469                 boiteDeDialogueAPropos.show();
00470                 break;
00471         }
00472
00473         return super.onOptionsItemSelected(item);
00474     }
```

8.5.2.22 onPause()

```
void com.lasalle.meeting.IHMMeeting.onPause ( ) [protected]
```

Méthode appelée après qu'une boîte de dialogue s'est affichée (on reprend sur un `onResume()`) ou avant `onStop()` (activité plus visible)

Définition à la ligne 174 du fichier `IHMMeeting.java`.

```
00175     {
00176         super.onPause();
00177         Log.d(TAG, "onPause()");
00178     }
```

8.5.2.23 onRadioButtonClicked()

```
void com.lasalle.meeting.IHMMeeting.onRadioButtonClicked (
    View vue )
```

Méthode appelée lorsque l'on coche un bouton radio.

Définition à la ligne 620 du fichier `IHMMeeting.java`.

Références `com.lasalle.meeting.EspaceDeTravail.INDICE_CHAUD`, `com.lasalle.meeting.EspaceDeTravail.INDICE_FRAIS`, `com.lasalle.meeting.EspaceDeTravail.INDICE_FROID`, `com.lasalle.meeting.EspaceDeTravail.INDICE_LEGEREMENT_FRAIS`, `com.lasalle.meeting.EspaceDeTravail.INDICE_LEGEREMENT_TIEDE`, `com.lasalle.meeting.EspaceDeTravail.INDICE_NEUTRE`, et `com.lasalle.meeting.EspaceDeTravail.INDICE_TIEDE`.

```
00621     {
00622         boolean estCoche = ((RadioButton) vue).isChecked();
00623
00624         switch (vue.getId())
00625         {
00626             case R.id.boutonRadioLibre:
00627                 if (estCoche)
00628                 {
00629                     estReserve = false;
00630                 }
00631                 break;
00632
00633             case R.id.boutonRadioOccupe:
00634                 if (estCoche)
00635                 {
00636                     estReserve = true;
00637                 }
00638                 break;
00639
00640             case R.id.boutonRadioChaud:
00641                 if (estCoche)
00642                 {
00643                     indiceDeConfort = EspaceDeTravail.INDICE_CHAUD;
00644                 }
00645                 break;
00646
00647             case R.id.boutonRadioTiede:
00648                 if (estCoche)
00649                 {
00650                     indiceDeConfort = EspaceDeTravail.INDICE_TIEDE;
00651                 }
00652                 break;
00653
00654             case R.id.boutonRadioLegerementTiede:
00655                 if (estCoche)
00656                 {
00657                     indiceDeConfort = EspaceDeTravail.INDICE_LEGEREMENT_TIEDE;
00658                 }
00659                 break;
00660
00661             case R.id.boutonRadioNeutre:
00662                 if (estCoche)
00663                 {
00664                     indiceDeConfort = EspaceDeTravail.INDICE_NEUTRE;
00665                 }
00666                 break;
00667
00668             case R.id.boutonRadioLegerementFrais:
00669                 if (estCoche)
00670                 {
00671                     indiceDeConfort = EspaceDeTravail.INDICE_LEGEREMENT_FRAIS;
00672                 }
00673                 break;
00674
00675             case R.id.boutonRadioFrais:
00676                 if (estCoche)
00677                 {
00678                     indiceDeConfort = EspaceDeTravail.INDICE_FRAIS;
00679                 }
00680                 break;
00681
00682             case R.id.boutonRadioFroid:
00683                 if (estCoche)
00684                 {
00685                     indiceDeConfort = EspaceDeTravail.INDICE_FROID;
00686                 }
00687                 break;
00688         }
00689     }
```

8.5.2.24 onResume()

```
void com.lasalle.meeting.IHMMeeting.onResume ( ) [protected]
```

Méthode appelée après [onStart\(\)](#) ou après [onPause\(\)](#)

Définition à la ligne [164](#) du fichier [IHMMeeting.java](#).

```
00165    {  
00166        super.onResume();  
00167        Log.d(TAG, "onResume()");  
00168    }
```

8.5.2.25 onStart()

```
void com.lasalle.meeting.IHMMeeting.onStart ( ) [protected]
```

Méthode appelée au démarrage de l'activité MainActivity.

Définition à la ligne [152](#) du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.IHMMeeting.initialiserEspacesDeTravail\(\)](#).

```
00153    {  
00154        super.onStart();  
00155        Log.d(TAG, "onStart()");  
00156  
00157        initialiserEspacesDeTravail();  
00158    }
```

8.5.2.26 onStop()

```
void com.lasalle.meeting.IHMMeeting.onStop ( ) [protected]
```

Méthode appelée lorsque l'activité n'est plus visible.

Définition à la ligne [184](#) du fichier [IHMMeeting.java](#).

```
00185    {  
00186        super.onStop();  
00187        Log.d(TAG, "onStop()");  
00188    }
```

8.5.2.27 rafraichirAffichageListeEspaces()

```
void com.lasalle.meeting.IHMMeeting.rafraichirAffichageListeEspaces ( ) [private]
```

Met à jour l'affichage de la liste.

Définition à la ligne 820 du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.IHMMeeting.trierEspacesDeTravail\(\)](#).

Référencé par [com.lasalle.meeting.IHMMeeting.filtrerParDisponibilite\(\)](#), [com.lasalle.meeting.IHMMeeting.filtrerParFavoris\(\)](#), [com.lasalle.meeting.IHMMeeting.filtrerParNiveauDeConfort\(\)](#), [com.lasalle.meeting.IHMMeeting.filtrerTous\(\)](#), et [com.lasalle.meeting.IHMMeeting.rechercher\(\)](#).

```
00821     {
00822         trierEspacesDeTravail ("nom",
00823     espacesDeTravailFiltres);
00824         swipeRefreshLayout.setRefreshing(false);
00825         adaptateur.notifyDataSetChanged();
00826     }
```

8.5.2.28 rafraichirListeEspacesFiltres()

```
void com.lasalle.meeting.IHMMeeting.rafraichirListeEspacesFiltres ( ) [private]
```

Fabrique la liste des espaces de travail à afficher.

Définition à la ligne 791 du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.IHMMeeting.filtrerParDisponibilite\(\)](#), [com.lasalle.meeting.IHMMeeting.filtrerParFavoris\(\)](#), [com.lasalle.meeting.IHMMeeting.filtrerParNiveauDeConfort\(\)](#), et [com.lasalle.meeting.IHMMeeting.filtrerTous\(\)](#).

Référencé par [com.lasalle.meeting.IHMMeeting.ajouterEspaceDeTravail\(\)](#), [com.lasalle.meeting.IHMMeeting.modifierEspaceDeTravail\(\)](#), et [com.lasalle.meeting.IHMMeeting.supprimerEspaceDeTravail\(\)](#).

```
00792     {
00793         switch(choixFiltre)
00794         {
00795             case R.id.actionAfficherTous:
00796                 titreEspacesDeTravail.setText("Tous les espaces de travail détectés");
00797                 filtrerTous();
00798                 break;
00799             case R.id.actionAfficherFavoris:
00800                 titreEspacesDeTravail.setText("Favoris");
00801                 filtrerParFavoris();
00802                 break;
00803             case R.id.actionFiltrerParDisponibilite:
00804                 titreEspacesDeTravail.setText("Espaces de travail filtrés par
00805 disponibilité");
00806                 filtrerParDisponibilite(estReserve);
00807                 break;
00808             case R.id.actionFiltrerParNiveauDeConfort:
00809                 titreEspacesDeTravail.setText("Espaces de travail filtrés par niveau
00810 de confort");
00811                 filtrerParNiveauDeConfort (
00812 indiceDeConfort);
00813                 break;
00814             }
00815         }
```

8.5.2.29 rechercher()

```
void com.lasalle.meeting.IHMMeeting.rechercher (
    String motCle ) [private]
```

Recherche les espaces de travail par un mot-clé

Définition à la ligne 694 du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.IHMMeeting.containsIgnoreCase\(\)](#), [com.lasalle.meeting.IHMMeeting.motCle](#), et [com.lasalle.meeting.IHMMeeting.rafraichirAffichageListeEspaces\(\)](#).

Référencé par [com.lasalle.meeting.IHMMeeting.initialiserBoiteDeDialogueRechercher\(\)](#).

```
00695     {
00696         espacesDeTravailFiltres.clear();
00697
00698         for(int i = 0; i < espacesDeTravail.size(); i++)
00699         {
00700             if( containsIgnoreCase(espacesDeTravail.elementAt(i).getNom()
, motCle) || containsIgnoreCase(espacesDeTravail.elementAt(i).
getDescription(), motCle) || containsIgnoreCase(
espacesDeTravail.elementAt(i).getLieu(), motCle) ||
containsIgnoreCase(espacesDeTravail.elementAt(i).getAdresseIP(),
motCle) )
00701             {
00702                 espacesDeTravailFiltres.add(
espacesDeTravail.elementAt(i));
00703             }
00704         }
00705
00706         Log.d(TAG, "rechercher() " + motCle);
00707
00708         rafraichirAffichageListeEspaces();
00709     }
```

8.5.2.30 recupererDonneesEspaceDeTravail()

```
static String com.lasalle.meeting.IHMMeeting.recupererDonneesEspaceDeTravail (
    EspaceDeTravail espaceDeTravail ) [static]
```

Récupère les données stockées d'un espace de travail.

Définition à la ligne 859 du fichier [IHMMeeting.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.setCode\(\)](#), [com.lasalle.meeting.EspaceDeTravail.setEstFavori\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.toJSON\(\)](#).

Référencé par [com.lasalle.meeting.EspaceDeTravail.EspaceDeTravail\(\)](#).

```
00860     {
00861         String donnees = "";
00862
00863         espaceDeTravail.setCode(preferences.getString(
PREFERENCES_CODE, null));
00864         espaceDeTravail.setEstFavori(preferences.getBoolean(
PREFERENCES_EST_FAVORI, false));
00865
00866         donnees = espaceDeTravail.toJSON();
00867
00868         return donnees;
00869     }
```

8.5.2.31 sauvegarderDonneesEspaceDeTravail()

```
static void com.lasalle.meeting.IHMMeeting.sauvegarderDonneesEspaceDeTravail (
    EspaceDeTravail espaceDeTravail ) [static]
```

Enregistre les données d'un espace de travail.

Définition à la ligne 874 du fichier `IHMMeeting.java`.

Références `com.lasalle.meeting.EspaceDeTravail.getCode()`, et `com.lasalle.meeting.EspaceDeTravail.getEstFavori()`.

Référencé par `com.lasalle.meeting.EspaceDeTravail.setEstFavori()`.

```
00875     {
00876         preferences.edit().putString(PREFERENCES_CODE, espaceDeTravail.getCode())
00877     }.apply();
00877         preferences.edit().putBoolean(PREFERENCES_EST_FAVORI,
    espaceDeTravail.getEstFavori()).apply();
00878     }
```

8.5.2.32 supprimerEspaceDeTravail()

```
void com.lasalle.meeting.IHMMeeting.supprimerEspaceDeTravail (
    EspaceDeTravail espaceDeTravail ) [private]
```

Supprime un espace de travail si déjà détecté

Paramètres

<i>espaceDeTravail</i>	l'espace de travail à supprimer
------------------------	---------------------------------

Définition à la ligne 255 du fichier `IHMMeeting.java`.

Références `com.lasalle.meeting.IHMMeeting.rafraichirListeEspacesFiltres()`, et `com.lasalle.meeting.IHMMeeting.verifierPresenceEspaceDeTravail()`.

```
00256     {
00257         int position = verifierPresenceEspaceDeTravail(espaceDeTravail);
00258
00259         if(position != -1)
00260         {
00261             espacesDeTravail.removeElementAt(position);
00262             rafraichirListeEspacesFiltres();
00263         }
00264     }
```

8.5.2.33 trierEspacesDeTravail()

```
void com.lasalle.meeting.IHMMeeting.trierEspacesDeTravail (
    final String champ,
    Vector< EspaceDeTravail > lesEspacesDeTravail ) [private]
```

Trie les espaces de travail.

Auteur

Thierry Vaira

Définition à la ligne 831 du fichier `IHMMeeting.java`.Références `com.lasalle.meeting.EspaceDeTravail.getAdresseIP()`, `com.lasalle.meeting.EspaceDeTravail.getNom()`, et `com.lasalle.meeting.EspaceDeTravail.getSuperficie()`.Référéncé par `com.lasalle.meeting.IHMMeeting.rafraichirAffichageListeEspaces()`.

```

00832     {
00833         //Log.d(TAG, "trierEspacesDeTravail() champ = " + champ + " - nb = " + lesEspacesDeTravail.size());
00834         Collections.sort(lesEspacesDeTravail, new Comparator<EspaceDeTravail>()
00835         {
00836             @Override
00837             public int compare(EspaceDeTravail e1, EspaceDeTravail e2)
00838             {
00839                 if(champ.equals("nom"))
00840                 {
00841                     return e1.getNom().compareTo(e2.getNom());
00842                 }
00843                 else if(champ.equals("adresseIP"))
00844                 {
00845                     return e1.getAdresseIP().compareTo(e2.getAdresseIP());
00846                 }
00847                 else if(champ.equals("superficie"))
00848                 {
00849                     return (e1.getSuperficie() - e2.getSuperficie());
00850                 }
00851                 return e1.getNom().compareTo(e2.getNom());
00852             }
00853         });
00854     }

```

8.5.2.34 `verifierPresenceEspaceDeTravail()` [1/2]

```

int com.lasalle.meeting.IHMMeeting.verifierPresenceEspaceDeTravail (
    EspaceDeTravail espaceDeTravail ) [private]

```

Vérifie la présence d'un espace de travail dans le conteneur des espaces de travail détectés.

Paramètres

<i>espaceDeTravail</i>	l'espace de travail à vérifier
------------------------	--------------------------------

Renvoie

int la position dans le conteneur sinon -1 en cas d'absence

Définition à la ligne 271 du fichier `IHMMeeting.java`.Références `com.lasalle.meeting.EspaceDeTravail.getNom()`.Référéncé par `com.lasalle.meeting.IHMMeeting.ajouterEspaceDeTravail()`, `com.lasalle.meeting.IHMMeeting.modifierEspaceDeTravail()`, et `com.lasalle.meeting.IHMMeeting.supprimerEspaceDeTravail()`.

```

00272     {
00273         for(int i = 0; i < espacesDeTravail.size(); ++i)
00274         {
00275             if(espaceDeTravail.getNom().equals(espacesDeTravail.elementAt(i).getNom()))
00276             {
00277                 return i;
00278             }
00279         }
00280         return -1;
00281     }

```

8.5.2.35 verifierPresenceEspaceDeTravail() [2/2]

```
int com.lasalle.meeting.IHMMeeting.verifierPresenceEspaceDeTravail (
    String adresseIP ) [private]
```

Vérifie la présence d'un espace de travail dans le conteneur des espaces de travail détectés.

Paramètres

<i>adresseIP</i> ↔	l'adresse IP d' un espace de travail à vérifier
--------------------	---

Renvoie

int la position dans le conteneur sinon -1 en cas d'absence

Définition à la ligne 288 du fichier [IHMMeeting.java](#).

```
00289    {
00290        for(int i = 0; i < espacesDeTravail.size(); ++i)
00291        {
00292            if(espacesDeTravail.elementAt(i).getAdresseIP().equals(adresseIP))
00293            {
00294                return i;
00295            }
00296        }
00297        return -1;
00298    }
```

8.5.3 Documentation des données membres

8.5.3.1 adaptateur

```
EspaceDeTravailAdaptateur com.lasalle.meeting.IHMMeeting.adaptateur [private]
```

Adaptateur pour les espaces de travail.

Définition à la ligne 73 du fichier [IHMMeeting.java](#).

8.5.3.2 boîteDeDialogueAPropos

```
AlertDialog.Builder com.lasalle.meeting.IHMMeeting.boiteDeDialogueAPropos [private]
```

Boîte de dialogue A propos.

Définition à la ligne 60 du fichier [IHMMeeting.java](#).

Référencé par [com.lasalle.meeting.IHMMeeting.initialiserBoitesDeDialogue\(\)](#), et [com.lasalle.meeting.IHMMeeting.onOptionsItemSelected\(\)](#)↔.

8.5.3.3 `boiteDeDialogueFiltrerParDisponibilite`

```
AlertDialog.Builder com.lasalle.meeting.IHMMeeting.boiteDeDialogueFiltrerParDisponibilite [private]
```

Boîte de dialogue Filtrer par disponibilité

Définition à la ligne 62 du fichier `IHMMeeting.java`.

Référencé par `com.lasalle.meeting.IHMMeeting.initialiserBoiteDeDialogueFiltrerParDisponibilite()`, et `com.lasalle.meeting.IHMMeeting.onOptionsItemSelected()`.

8.5.3.4 `boiteDeDialogueFiltrerParNiveauDeConfort`

```
AlertDialog.Builder com.lasalle.meeting.IHMMeeting.boiteDeDialogueFiltrerParNiveauDeConfort [private]
```

Boîte de dialogue Filtrer par niveau de confort.

Définition à la ligne 63 du fichier `IHMMeeting.java`.

Référencé par `com.lasalle.meeting.IHMMeeting.initialiserBoiteDeDialogueFiltrerParNiveauDeConfort()`, et `com.lasalle.meeting.IHMMeeting.onOptionsItemSelected()`.

8.5.3.5 `boiteDeDialogueRechercher`

```
AlertDialog com.lasalle.meeting.IHMMeeting.boiteDeDialogueRechercher [private]
```

Boîte de dialogue Rechercher.

Définition à la ligne 61 du fichier `IHMMeeting.java`.

Référencé par `com.lasalle.meeting.IHMMeeting.onOptionsItemSelected()`.

8.5.3.6 `choixFiltre`

```
int com.lasalle.meeting.IHMMeeting.choixFiltre = -1 [private]
```

Choix du dernier type de filtrage.

Définition à la ligne 76 du fichier `IHMMeeting.java`.

8.5.3.7 `communication`

```
Communication com.lasalle.meeting.IHMMeeting.communication = null [static], [private]
```

Attribut permettant d'envoyer des requêtes.

Définition à la ligne 74 du fichier `IHMMeeting.java`.

8.5.3.8 espacesDeTravail

```
Vector<EspaceDeTravail> com.lasalle.meeting.IHMMeeting.espacesDeTravail [static], [private]
```

Conteneur pour les espaces de travail.

Les attributs

Définition à la ligne 71 du fichier [IHMMeeting.java](#).

8.5.3.9 espacesDeTravailFiltres

```
Vector<EspaceDeTravail> com.lasalle.meeting.IHMMeeting.espacesDeTravailFiltres [static], [private]
```

Conteneur pour les espaces de travail filtrés.

Définition à la ligne 72 du fichier [IHMMeeting.java](#).

8.5.3.10 estReserve

```
boolean com.lasalle.meeting.IHMMeeting.estReserve [private]
```

Définition à la ligne 65 du fichier [IHMMeeting.java](#).

8.5.3.11 handler

```
Handler com.lasalle.meeting.IHMMeeting.handler [private]
```

Permet de récupérer les trames.

Paramètres

<i>msg</i>	message
------------	---------

Définition à la ligne 360 du fichier [IHMMeeting.java](#).

Référencé par [com.lasalle.meeting.IHMMeeting.demarrerReseau\(\)](#).

8.5.3.12 indiceDeConfort

```
int com.lasalle.meeting.IHMMeeting.indiceDeConfort [private]
```

Définition à la ligne 66 du fichier [IHMMeeting.java](#).

Référencé par [com.lasalle.meeting.IHMMeeting.filtrerParNiveauDeConfort\(\)](#).

8.5.3.13 `listeEspacesDeTravail`

```
ListView com.lasalle.meeting.IHMMeeting.listeEspacesDeTravail [private]
```

Affichage des espaces de travail sous forme de liste.

Définition à la ligne 58 du fichier [IHMMeeting.java](#).

8.5.3.14 `motCle`

```
String com.lasalle.meeting.IHMMeeting.motCle [private]
```

Mot-clé pour rechercher un espace de travail.

Définition à la ligne 64 du fichier [IHMMeeting.java](#).

Référencé par [com.lasalle.meeting.IHMMeeting.rechercher\(\)](#).

8.5.3.15 `PREFERENCES`

```
final String com.lasalle.meeting.IHMMeeting.PREFERENCES = "preferences" [static], [private]
```

Préférences de l'application.

Définition à la ligne 50 du fichier [IHMMeeting.java](#).

8.5.3.16 `preferences`

```
SharedPreferences com.lasalle.meeting.IHMMeeting.preferences [static], [private]
```

Pour le stockage de données.

Définition à la ligne 77 du fichier [IHMMeeting.java](#).

8.5.3.17 `PREFERENCES_CODE`

```
final String com.lasalle.meeting.IHMMeeting.PREFERENCES_CODE = "code" [static], [private]
```

Code pour libérer l'espace de travail enregistré dans les préférences.

Définition à la ligne 51 du fichier [IHMMeeting.java](#).

8.5.3.18 PREFERENCES_EST_FAVORI

```
final String com.lasalle.meeting.IHMMeeting.PREFERENCES_EST_FAVORI = "false" [static], [private]
```

Favori enregistré dans les préférences.

Définition à la ligne 52 du fichier [IHMMeeting.java](#).

8.5.3.19 swipeRefreshLayout

```
SwipeRefreshLayout com.lasalle.meeting.IHMMeeting.swipeRefreshLayout [private]
```

Pour le Pull To Refresh.

Les ressources de l'IHM

Définition à la ligne 57 du fichier [IHMMeeting.java](#).

8.5.3.20 TAG

```
final String com.lasalle.meeting.IHMMeeting.TAG = "_IHMMeeting" [static], [private]
```

TAG pour les logs.

Les constantes

Définition à la ligne 49 du fichier [IHMMeeting.java](#).

8.5.3.21 titreEspacesDeTravail

```
TextView com.lasalle.meeting.IHMMeeting.titreEspacesDeTravail [private]
```

Affichage du titre principal.

Définition à la ligne 59 du fichier [IHMMeeting.java](#).

8.5.3.22 wm

```
WifiManager com.lasalle.meeting.IHMMeeting.wm = null [private]
```

Attribut permettant de voir la connexion au WiFi.

Définition à la ligne 75 du fichier [IHMMeeting.java](#).

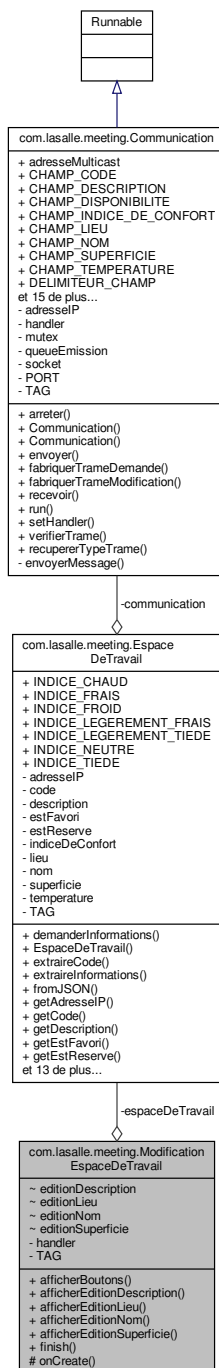
La documentation de cette classe a été générée à partir du fichier suivant :

— [IHMMeeting.java](#)

8.6 Référence de la classe com.lasalle.meeting.ModificationEspaceDeTravail

L'activité de modification d'un espace de travail de l'application Meeting.

Graphe de collaboration de com.lasalle.meeting.ModificationEspaceDeTravail :



Fonctions membres publiques

- void `afficherBoutons()`
Affiche le bouton "Enregistrer".
- void `afficherEditionDescription()`

- void [afficherEditionLieu](#) ()
Affiche la zone d'édition de la description de l'espace de travail.
- void [afficherEditionNom](#) ()
Affiche la zone d'édition du lieu de l'espace de travail.
- void [afficherEditionSuperficie](#) ()
Affiche la zone d'édition du nom de l'espace de travail.
- void [finish](#) ()
Affiche la zone d'édition de la superficie de l'espace de travail.
- void [finish](#) ()
Termine l'activité de modification d'un espace de travail.

Fonctions membres protégées

- void [onCreate](#) (Bundle savedInstanceState)
Méthode appelée à la création de l'activité

Attributs privés

- [EspaceDeTravail](#) [espaceDeTravail](#)
L'espace de travail.
- Handler [handler](#)
Permet de récupérer les trames.

Attributs privés statiques

- static final String [TAG](#) = "_ModificationEspaceDeTravail"
TAG pour les logs.

8.6.1 Description détaillée

L'activité de modification d'un espace de travail de l'application Meeting.

Définition à la ligne 30 du fichier [ModificationEspaceDeTravail.java](#).

8.6.2 Documentation des fonctions membres

8.6.2.1 afficherBoutons()

```
void com.lasalle.meeting.ModificationEspaceDeTravail.afficherBoutons ( )
```

Affiche le bouton "Enregistrer".

Définition à la ligne 116 du fichier [ModificationEspaceDeTravail.java](#).

Références [com.lasalle.meeting.ModificationEspaceDeTravail.finish\(\)](#), et [com.lasalle.meeting.EspaceDeTravail.modifierInformations\(\)](#).

Référencé par [com.lasalle.meeting.ModificationEspaceDeTravail.onCreate\(\)](#).

```
00117     {
00118         Button boutonEnregistrer = (Button) findViewById(R.id.boutonEnregistrer);
00119
00120         boutonEnregistrer.setOnClickListener(
00121             new View.OnClickListener()
00122             {
00123                 public void onClick(View v)
00124                 {
00125                     String champs[] = new String[] { editionNom.getText().toString(), editionDescription.
00126 getText().toString(), editionLieu.getText().toString(), editionSuperficie.getText().toString() };
00127                     List<String> parametres = Arrays.asList(champs);
00128                     espaceDeTravail.modifierInformations(parametres);
00129
00130                     finish();
00131                 }
00132             }
00133         );
00134     }
```


8.6.2.2 `afficherEditionDescription()`

```
void com.lasalle.meeting.ModificationEspaceDeTravail.afficherEditionDescription ( )
```

Affiche la zone d'édition de la description de l'espace de travail.

Définition à la ligne 93 du fichier `ModificationEspaceDeTravail.java`.

Références `com.lasalle.meeting.EspaceDeTravail.getDescription()`.

Référencé par `com.lasalle.meeting.ModificationEspaceDeTravail.onCreate()`.

```
00094    {
00095        editionDescription = (EditText) findViewById(R.id.editionDescription);
00096        editionDescription.setText(espaceDeTravail.getDescription());
00097
00098        Log.d(TAG, "afficherEditionDescription() " + espaceDeTravail.
00099            getDescription());
00099    }
```

8.6.2.3 `afficherEditionLieu()`

```
void com.lasalle.meeting.ModificationEspaceDeTravail.afficherEditionLieu ( )
```

Affiche la zone d'édition du lieu de l'espace de travail.

Définition à la ligne 82 du fichier `ModificationEspaceDeTravail.java`.

Références `com.lasalle.meeting.EspaceDeTravail.getLieu()`.

Référencé par `com.lasalle.meeting.ModificationEspaceDeTravail.onCreate()`.

```
00083    {
00084        editionLieu = (EditText) findViewById(R.id.editionLieu);
00085        editionLieu.setText(espaceDeTravail.getLieu());
00086
00087        Log.d(TAG, "afficherEditionLieu() " + espaceDeTravail.
00088            getLieu());
00088    }
```

8.6.2.4 `afficherEditionNom()`

```
void com.lasalle.meeting.ModificationEspaceDeTravail.afficherEditionNom ( )
```

Affiche la zone d'édition du nom de l'espace de travail.

Définition à la ligne 72 du fichier `ModificationEspaceDeTravail.java`.

Références `com.lasalle.meeting.EspaceDeTravail.getNom()`.

Référencé par `com.lasalle.meeting.ModificationEspaceDeTravail.onCreate()`.

```
00073    {
00074        editionNom = (EditText) findViewById(R.id.editionNom);
00075        editionNom.setText(espaceDeTravail.getNom());
00076
00077        Log.d(TAG, "afficherEditionNom() " + espaceDeTravail.
00078            getNom());
00078    }
```

8.6.2.5 afficherEditionSuperficie()

```
void com.lasalle.meeting.ModificationEspaceDeTravail.afficherEditionSuperficie ( )
```

Affiche la zone d'édition de la superficie de l'espace de travail.

Définition à la ligne 104 du fichier [ModificationEspaceDeTravail.java](#).

Références [com.lasalle.meeting.EspaceDeTravail.getSuperficie\(\)](#).

Référencé par [com.lasalle.meeting.ModificationEspaceDeTravail.onCreate\(\)](#).

```
00105      {
00106          editionSuperficie = (EditText) findViewById(R.id.editionSuperficie);
00107          int superficie = espaceDeTravail.getSuperficie\(\);
00108          editionSuperficie.setText (String.valueOf(superficie));
00109
00110          Log.d(TAG, "afficherEditionSuperficie() " + espaceDeTravail.
getSuperficie\(\));
00111      }
```

8.6.2.6 finish()

```
void com.lasalle.meeting.ModificationEspaceDeTravail.finish ( )
```

Termine l'activité de modification d'un espace de travail.

Définition à la ligne 139 du fichier [ModificationEspaceDeTravail.java](#).

Référencé par [com.lasalle.meeting.ModificationEspaceDeTravail.afficherBoutons\(\)](#).

```
00140      {
00141          Log.d(TAG, "finish()");
00142
00143          Intent intent = new Intent();
00144          //intent.putExtra("unEspaceDeTravail", espaceDeTravail);
00145          setResult (RESULT_OK, intent);
00146          super.finish();
00147      }
```

8.6.2.7 onCreate()

```
void com.lasalle.meeting.ModificationEspaceDeTravail.onCreate (
    Bundle savedInstanceState ) [protected]
```

Méthode appelée à la création de l'activité

Définition à la ligne 54 du fichier [ModificationEspaceDeTravail.java](#).

Références [com.lasalle.meeting.ModificationEspaceDeTravail.afficherBoutons\(\)](#), [com.lasalle.meeting.ModificationEspaceDeTravail.afficherEditionDescription\(\)](#), [com.lasalle.meeting.ModificationEspaceDeTravail.afficherEditionLieu\(\)](#), [com.lasalle.meeting.ModificationEspaceDeTravail.afficherEditionNom\(\)](#), [com.lasalle.meeting.ModificationEspaceDeTravail.afficherEditionSuperficie\(\)](#), [com.lasalle.meeting.ModificationEspaceDeTravail.handler](#), et [com.lasalle.meeting.EspaceDeTravail.initialiserCommunication\(\)](#).

```
00055      {
00056          super.onCreate(savedInstanceState);
00057          setContentView(R.layout.activity_modification_espace_de_travail);
00058          Intent intent = getIntent();
00059          espaceDeTravail = (EspaceDeTravail) intent.getSerializableExtra ("unEspaceDeTravail");
00060          espaceDeTravail.initialiserCommunication\(
handler\);
00061
00062          afficherEditionNom\(\);
00063          afficherEditionLieu\(\);
00064          afficherEditionDescription\(\);
00065          afficherEditionSuperficie\(\);
00066          afficherBoutons\(\);
00067      }
```

8.6.3 Documentation des données membres

8.6.3.1 espaceDeTravail

`EspaceDeTravail` com.lasalle.meeting.ModificationEspaceDeTravail.espaceDeTravail [private]

L'espace de travail.

Les attributs

Définition à la ligne 48 du fichier `ModificationEspaceDeTravail.java`.

8.6.3.2 handler

`Handler` com.lasalle.meeting.ModificationEspaceDeTravail.handler [private]

Valeur initiale :

```
= new Handler()
{
    @Override
    public void handleMessage(Message msg)
    {
        super.handleMessage(msg);
        Bundle b = msg.getData();

        switch(msg.what)
        {
            case Communication.TYPE_RECEPTION:
                String trame = b.getString("donnees");
                Log.d(TAG, "handleMessage() Réception [" + b.getString("adresseIP") + ":" + b.getInt("port") + "] -> " + trame);

                String[] champs = trame.split(";");
                int typeTrame = Communication.recupererTypeTrame(champs);
                Log.d(TAG, "handleMessage() : typeTrame : " + typeTrame);

                break;
            default:
                Log.d(TAG, "handleMessage() : code inconnu ! ");
        }
    }
}
```

Permet de récupérer les trames.

Paramètres

<code>msg</code>	message
------------------	---------

Définition à la ligne 153 du fichier `ModificationEspaceDeTravail.java`.

Référencé par `com.lasalle.meeting.ModificationEspaceDeTravail.onCreate()`.

8.6.3.3 TAG

```
final String com.lasalle.meeting.ModificationEspaceDeTravail.TAG = "_ModificationEspaceDeTravail" [static],
[private]
```

TAG pour les logs.

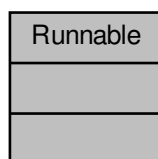
Les constantes

Définition à la ligne 35 du fichier [ModificationEspaceDeTravail.java](#).

La documentation de cette classe a été générée à partir du fichier suivant :
— [ModificationEspaceDeTravail.java](#)

8.7 Référence de la classe Runnable

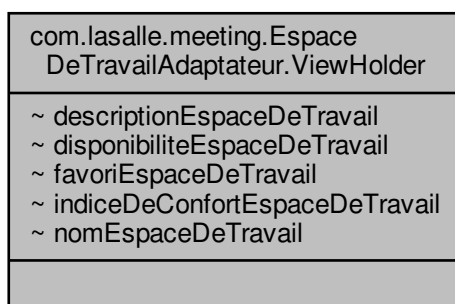
Graphe de collaboration de Runnable :



La documentation de cette classe a été générée à partir du fichier suivant :
— [Communication.java](#)

8.8 Référence de la classe com.lasalle.meeting.EspaceDeTravailAdaptateur.ViewHolder

Graphe de collaboration de com.lasalle.meeting.EspaceDeTravailAdaptateur.ViewHolder :



8.8.1 Description détaillée

Définition à la ligne 37 du fichier [EspaceDeTravailAdaptateur.java](#).

La documentation de cette classe a été générée à partir du fichier suivant :
— [EspaceDeTravailAdaptateur.java](#)

9 Documentation des fichiers

9.1 Référence du fichier AffichageEspaceDeTravail.java

Déclaration de la classe AffichageEspaceDeTravail.

Classes

- class [com.lasalle.meeting.AffichageEspaceDeTravail](#)
L'activité d'affichage d'un espace de travail de l'application Meeting.

Paquetages

- package [com.lasalle.meeting](#)

9.1.1 Description détaillée

Déclaration de la classe AffichageEspaceDeTravail.

Auteur

KELLER-LAVALLEE Joachim

LastChangedRevision

93

LastChangedDate

2021-06-10 16 :36 :23 +0200 (jeu. 10 juin 2021)

Définition dans le fichier [AffichageEspaceDeTravail.java](#).

9.2 AffichageEspaceDeTravail.java

```
00001 package com.lasalle.meeting;
00002
00003 import androidx.appcompat.app.AppCompatActivity;
00004
00005 import android.app.AlertDialog;
00006 import android.content.DialogInterface;
00007 import android.content.Intent;
00008 import android.graphics.Color;
00009 import android.os.Bundle;
00010 import android.os.Handler;
00011 import android.os.Message;
00012 import android.util.Log;
00013 import android.view.LayoutInflater;
00014 import android.view.View;
00015 import android.widget.EditText;
00016 import android.widget.ImageView;
00017 import android.widget.TextView;
00018 import android.widget.Button;
00019
00020 import java.io.Serializable;
00021
00034 public class AffichageEspaceDeTravail extends AppCompatActivity
00035 {
00039     private static final String TAG = "_AffichageEspaceTravail";
00040
00044     private EspaceDeTravail espaceDeTravail;
00045
00049     @Override
```

```

00050     protected void onCreate(Bundle savedInstanceState)
00051     {
00052         super.onCreate(savedInstanceState);
00053         setContentView(R.layout.activity_affichage_espace_de_travail);
00054         Intent intent = getIntent();
00055         espaceDeTravail = (EspaceDeTravail) intent.getSerializableExtra("unEspaceDeTravail");
00056         espaceDeTravail.initialiserCommunication(handler);
00057
00058         afficher();
00059     }
00060
00064     private void afficher()
00065     {
00066         afficherAdresseIP();
00067         afficherNom();
00068         afficherLieu();
00069         afficherDescription();
00070         afficherSuperficie();
00071         afficherTemperature();
00072         afficherIndiceDeConfort();
00073         afficherDisponibilite();
00074         afficherFavori();
00075         afficherBoutons();
00076     }
00077
00081     public void afficherAdresseIP()
00082     {
00083         TextView affichageAdresseIP = (TextView) findViewById(R.id.affichageAdresseIP);
00084         affichageAdresseIP.setText(espaceDeTravail.getAdresseIP());
00085
00086         Log.d(TAG, "afficherAdresseIP() " + espaceDeTravail.getAdresseIP());
00087     }
00088
00092     public void afficherNom()
00093     {
00094         TextView affichageNom = (TextView) findViewById(R.id.affichageNom);
00095         affichageNom.setText(espaceDeTravail.getNom());
00096
00097         Log.d(TAG, "afficherNom() " + espaceDeTravail.getNom());
00098     }
00099
00103     public void afficherLieu()
00104     {
00105         TextView affichageLieu = (TextView) findViewById(R.id.affichageLieu);
00106         affichageLieu.setText(espaceDeTravail.getLieu());
00107
00108         Log.d(TAG, "afficherLieu() " + espaceDeTravail.getLieu());
00109     }
00110
00114     public void afficherDescription()
00115     {
00116         TextView affichageDescription = (TextView) findViewById(R.id.affichageDescription);
00117         affichageDescription.setText(espaceDeTravail.getDescription());
00118
00119         Log.d(TAG, "afficherDescription() " + espaceDeTravail.getDescription());
00120     }
00121
00125     public void afficherSuperficie()
00126     {
00127         TextView affichageSuperficie = (TextView) findViewById(R.id.affichageSuperficie);
00128         affichageSuperficie.setText(String.valueOf(espaceDeTravail.getSuperficie()));
00129
00130         Log.d(TAG, "afficherSuperficie() " + espaceDeTravail.getSuperficie());
00131     }
00132
00136     public void afficherTemperature()
00137     {
00138         TextView affichageTemperature = (TextView) findViewById(R.id.affichageTemperature);
00139         affichageTemperature.setText(String.valueOf(espaceDeTravail.
00140 getTemperature()));
00141
00142         Log.d(TAG, "afficherTemperature() " + espaceDeTravail.getTemperature());
00143     }
00144
00147     public void afficherIndiceDeConfort()
00148     {
00149         TextView affichageIndiceDeConfort = (TextView) findViewById(R.id.affichageIndiceDeConfort);
00150
00151         switch(espaceDeTravail.getIndiceDeConfort())
00152         {
00153             case EspaceDeTravail.INDICE_CHAUD:
00154                 affichageIndiceDeConfort.setText("Chaud");
00155                 break;
00156
00157             case EspaceDeTravail.INDICE_TIEDE:
00158                 affichageIndiceDeConfort.setText("Tiède");
00159                 break;
00160
00161             case EspaceDeTravail.INDICE_LEGEREMENT_TIEDE:
00162                 affichageIndiceDeConfort.setText("Légèrement tiède");
00163                 break;

```

```

00164
00165         case EspaceDeTravail.INDICE_NEUTRE:
00166             affichageIndiceDeConfort.setText("Neutre");
00167             break;
00168
00169         case EspaceDeTravail.INDICE_LEGEREMENT_FRAIS:
00170             affichageIndiceDeConfort.setText("Légèrement frais");
00171             break;
00172
00173         case EspaceDeTravail.INDICE_FRAIS:
00174             affichageIndiceDeConfort.setText("Frais");
00175             break;
00176
00177         case EspaceDeTravail.INDICE_FROID:
00178             affichageIndiceDeConfort.setText("Froid");
00179             break;
00180     }
00181
00182     Log.d(TAG, "afficherIndiceDeConfort() " + espaceDeTravail.
00183         getIndiceDeConfort());
00184
00185     public void afficherFavori()
00186     {
00187         ImageView iconeFavori = (ImageView) findViewById(R.id.iconeFavori);
00188
00189         if(!espaceDeTravail.getEstFavori())
00190         {
00191             Log.d(TAG, "afficherFavori() Non favori");
00192             iconeFavori.setVisibility(View.INVISIBLE);
00193         }
00194         else
00195         {
00196             Log.d(TAG, "afficherFavori() Favori");
00197             iconeFavori.setVisibility(View.VISIBLE);
00198         }
00199
00200         Log.d(TAG, "afficherFavori() " + espaceDeTravail.getEstFavori());
00201     }
00202
00203     public void afficherDisponibilite()
00204     {
00205         TextView affichageDisponibilite = (TextView) findViewById(R.id.affichageDisponibilite);
00206
00207         if(!espaceDeTravail.getEstReserve())
00208         {
00209             Log.d(TAG, "afficherDisponibilite() Libre");
00210             affichageDisponibilite.setText("Libre");
00211             affichageDisponibilite.setTextColor(Color.parseColor("#00FF00")); // Color.rgb(0,255,0)
00212         }
00213         else
00214         {
00215             Log.d(TAG, "afficherDisponibilite() Occupé");
00216             affichageDisponibilite.setText("Occupé");
00217             affichageDisponibilite.setTextColor(Color.rgb(255,0,0));
00218         }
00219
00220         Log.d(TAG, "afficherDisponibilite() " + espaceDeTravail.getEstReserve());
00221     }
00222
00223     public void afficherBoutons()
00224     {
00225         Button boutonReserver = (Button) findViewById(R.id.boutonReserver);
00226         Button boutonLiberer = (Button) findViewById(R.id.boutonLiberer);
00227         Button boutonModifier = (Button) findViewById(R.id.boutonModifier);
00228         Button boutonAjouterFavori = (Button) findViewById(R.id.boutonAjouterFavori);
00229         Button boutonRetirerFavori = (Button) findViewById(R.id.boutonRetirerFavori);
00230
00231         boutonReserver.setOnClickListener(
00232             new View.OnClickListener()
00233             {
00234                 public void onClick(View v)
00235                 {
00236                     espaceDeTravail.reserver();
00237                     afficherDisponibilite();
00238                     afficherBoutons();
00239                 }
00240             }
00241         );
00242
00243         boutonLiberer.setOnClickListener(
00244             new View.OnClickListener()
00245             {
00246                 public void onClick(View v)
00247                 {
00248                     afficherBoiteLiberation();
00249                     afficherDisponibilite();
00250                     afficherBoutons();
00251                 }
00252             }
00253         );
00254     }
00255
00256
00257
00258
00259
00260
00261
00262

```

```

00263
00264         if(!espaceDeTravail.getEstReserve())
00265         {
00266             boutonReserver.setVisibility(View.VISIBLE);
00267             boutonLiberer.setVisibility(View.GONE);
00268         }
00269         else
00270         {
00271             boutonReserver.setVisibility(View.GONE);
00272             boutonLiberer.setVisibility(View.VISIBLE);
00273         }
00274
00275         boutonModifier.setOnClickListener(
00276             new View.OnClickListener()
00277             {
00278                 public void onClick(View v)
00279                 {
00280                     Intent intent = new Intent(AffichageEspaceDeTravail.this,
ModificationEspaceDeTravail.class);
00281                     espaceDeTravail.initialiserCommunication(null);
00282                     intent.putExtra("unEspaceDeTravail", (Serializable)espaceDeTravail);
00283                     startActivityForResult(intent, 0);
00284                 }
00285             }
00286         );
00287
00288         boutonAjouterFavori.setOnClickListener(
00289             new View.OnClickListener()
00290             {
00291                 public void onClick(View v)
00292                 {
00293                     espaceDeTravail.setEstFavori(true);
00294                     afficherFavori();
00295                     afficherBoutons();
00296                 }
00297             }
00298         );
00299
00300         boutonRetirerFavori.setOnClickListener(
00301             new View.OnClickListener()
00302             {
00303                 public void onClick(View v)
00304                 {
00305                     espaceDeTravail.setEstFavori(false);
00306                     afficherFavori();
00307                     afficherBoutons();
00308                 }
00309             }
00310         );
00311
00312         if(!espaceDeTravail.getEstFavori())
00313         {
00314             boutonAjouterFavori.setVisibility(View.VISIBLE);
00315             boutonRetirerFavori.setVisibility(View.GONE);
00316         }
00317         else
00318         {
00319             boutonAjouterFavori.setVisibility(View.GONE);
00320             boutonRetirerFavori.setVisibility(View.VISIBLE);
00321         }
00322     }
00323
00327     public void afficherBoiteLiberation()
00328     {
00329         AlertDialog.Builder boiteLiberation = new AlertDialog.Builder(this);
00330         boiteLiberation.setTitle("Libérer l'espace de travail");
00331         boiteLiberation.setMessage("Saisissez le code pour libérer l'espace de travail :");
00332         LayoutInflater inflater = this.getLayoutInflater();
00333         View vue = inflater.inflate(R.layout.boite_liberation, null);
00334         boiteLiberation.setView(vue);
00335
00336         boiteLiberation.setPositiveButton("Libérer", new DialogInterface.OnClickListener()
00337         {
00338             public void onClick(DialogInterface dialog, int which)
00339             {
00340                 EditText saisieCode = (EditText) ((AlertDialog) dialog).findViewById(R.id.saisieCode);
00341                 Log.d(TAG, "onClick() code = " + saisieCode.getText().toString());
00342                 espaceDeTravail.liberer(saisieCode.getText().toString());
00343             }
00344         });
00345         boiteLiberation.setNegativeButton("Annuler", new DialogInterface.OnClickListener()
00346         {
00347             public void onClick(DialogInterface dialog, int which)
00348             {
00349             }
00350         });
00351     });
00352
00353     boiteLiberation.show();
00354 }
00355

```



```

00359     @Override
00360     protected void onActivityResult(int requestCode, int resultCode, Intent data)
00361     {
00362         super.onActivityResult(requestCode, resultCode, data);
00363         Log.d(TAG, "onActivityResult() requestCode=" + requestCode + " - resultCode=" + resultCode + "");
00364
00365         finish();
00366     }
00367
00371     @Override
00372     public void finish()
00373     {
00374         Log.d(TAG, "finish()");
00375
00376         Intent intent = new Intent();
00377         //intent.putExtra("unEspaceDeTravail", espaceDeTravail);
00378         setResult(RESULT_OK, intent);
00379         super.finish();
00380     }
00381
00382     private Handler handler = new Handler() {
00383         @Override
00384         public void handleMessage(Message msg)
00385         {
00386             super.handleMessage(msg);
00387             Bundle b = msg.getData();
00388             Log.d(TAG, "handleMessage() " + b);
00389
00390             switch(msg.what)
00391             {
00392                 case Communication.TYPE_RECEPTION:
00393                     String trame = b.getString("donnees");
00394                     Log.d(TAG, "handleMessage() Réception [" + b.getString("adresseIP") + ":" + b.getInt("
00395 port") + "] -> " + trame);
00396
00397                     String[] champs = trame.split(";");
00398                     int typeTrame = Communication.
00399 recupererTypeTrame(champs);
00400
00401                     switch(typeTrame)
00402                     {
00403                         case Communication.
00404 MODIFICATION_DISPONIBILITE:
00405                             if(espaceDeTravail.getAdresseIP().equals(b.getString("adresseIP")))
00406                             {
00407                                 if(espaceDeTravail.extraireCode(trame))
00408                                 {
00409                                     if(espaceDeTravail.getCode().isEmpty())
00410                                         espaceDeTravail.setEstReserve(false);
00411                                     else
00412                                         espaceDeTravail.setEstReserve(true);
00413                                 }
00414                                 afficherDisponibilite();
00415                                 afficherBoutons();
00416                             }
00417                             break;
00418                         case Communication.
00419 MODIFICATION_INFORMATIONS:
00420                             if(espaceDeTravail.getAdresseIP().equals(b.getString("adresseIP")))
00421                             {
00422                                 if(espaceDeTravail.extraireInformations(trame))
00423                                 {
00424                                     espaceDeTravail.extraireInformations(trame);
00425                                 }
00426                                 afficherNom();
00427                                 afficherDescription();
00428                                 afficherLieu();
00429                                 afficherSuperficie();
00430                                 afficherDisponibilite();
00431                                 afficherBoutons();
00432                             }
00433                             break;
00434                         default:
00435                             Log.d(TAG, "handleMessage() : type de trame inconnu !");
00436                             break;
00437                     }
00438                 default:
00439                     Log.d(TAG, "handleMessage() : code inconnu !");
00440             }
00441         }
00442     };

```

9.3 Référence du fichier Communication.java

Déclaration de la classe Communication.

Classes

- class [com.lasalle.meeting.Communication](#)
Communication entre l'application et le portier.

Paquetages

- package [com.lasalle.meeting](#)

9.3.1 Description détaillée

Déclaration de la classe Communication.

Auteur

KELLER-LAVALLEE Joachim

LastChangedRevision

94

LastChangedDate

2021-06-11 12 :16 :56 +0200 (ven. 11 juin 2021)

Définition dans le fichier [Communication.java](#).

9.4 Communication.java

```
00001 package com.lasalle.meeting;
00002
00003 import android.os.Bundle;
00004 import android.os.Handler;
00005 import android.os.Message;
00006 import android.util.Log;
00007
00008 import java.io.IOException;
00009 import java.net.DatagramPacket;
00010 import java.net.DatagramSocket;
00011 import java.net.InetAddress;
00012 import java.net.UnknownHostException;
00013 import java.util.List;
00014 import java.util.concurrent.LinkedBlockingQueue;
00015 import java.util.concurrent.locks.ReentrantLock;
00016
00029 public class Communication implements Runnable
00030 {
00034     private static final String TAG = "_Communication";
00035
00039     private InetAddress adresseIP = null;
00040     public final static String adresseMulticast = "239.0.0.42";
00041     private final static int PORT = 5000;
00042     public final static int TYPE_RECEPTION = 1;
00043     private final ReentrantLock mutex = new ReentrantLock();
00044     private DatagramSocket socket = null;
00045     private LinkedBlockingQueue<DatagramPacket> queueEmission;
00046     private Handler handler;
00047
00051     public static final String DELIMITEUR_EN_TETE = "$";
00052     public static final String DELIMITEUR_CHAMP = ";";
00053     public static final String DELIMITEUR_FIN = "\r\n";
00054     public static final int TRAME_INCONNUE = -1;
```

```

00055     public static final int DEMANDE_INFORMATIONS = 1;
00056     public static final int DEMANDE_DISPONIBILITE = 3;
00057     public static final int MODIFICATION_INFORMATIONS = 1;
00058     public static final int MODIFICATION_DISPONIBILITE = 3;
00059     public static final int NB_CHAMPS_INFORMATIONS = 4;
00060     public static final int NB_CHAMPS_DISPONIBILITE = 1;
00061     public static final int NB_CHAMPS_DISPONIBILITE_CODE = 2;
00062     public static final int NB_CHAMPS_DEMANDE_INFORMATIONS = 7;
00063     public static final int NB_CHAMPS_DEMANDE_DISPONIBILITE = 1;
00064     public static final int NB_CHAMPS_MODIFICATION_DISPONIBILITE = 2;
00065     public static final int NB_CHAMPS_RETOUR_MODIFICATION_DISPONIBILITE
= 3;
00066     public static final int CHAMP_NOM = 0;
00067     public static final int CHAMP_DESCRIPTION = 1;
00068     public static final int CHAMP_LIEU = 2;
00069     public static final int CHAMP_SUPERFICIE = 3;
00070     public static final int CHAMP_DISPONIBILITE = 4;
00071     public static final int CHAMP_INDICE_DE_CONFORT = 5;
00072     public static final int CHAMP_TEMPERATURE = 6;
00073     public static final int CHAMP_CODE = 1;
00074
00079     public Communication(Handler handler)
00080     {
00081         this.handler = handler;
00082
00083         try
00084         {
00085             socket = new DatagramSocket(PORT);
00086             Log.d(TAG, "Création de la socket UDP sur le port " + PORT);
00087         }
00088         catch (Exception e)
00089         {
00090             e.printStackTrace();
00091         }
00092
00093         queueEmission = new LinkedBlockingQueue<DatagramPacket>();
00094     }
00095
00096     public Communication()
00097     {
00098         this.handler = null;
00099
00100         try
00101         {
00102             socket = new DatagramSocket();
00103             Log.d(TAG, "Création d'une socket UDP");
00104         }
00105         catch (Exception e)
00106         {
00107             e.printStackTrace();
00108         }
00109
00110         queueEmission = new LinkedBlockingQueue<DatagramPacket>();
00111     }
00112
00113     public void setHandler(Handler handler)
00114     {
00115         this.handler = handler;
00116     }
00117
00123     public void envoyer(String trame, String adressePortier)
00124     {
00125         if(socket == null || socket.isClosed())
00126             return;
00127
00128         Log.d(TAG, "envoyer() : adressePortier = " + adressePortier);
00129
00130         final InetAddress adresseIPDistante;
00131         try
00132         {
00133             adresseIPDistante = InetAddress.getByName(adressePortier);
00134         }
00135         catch (UnknownHostException e)
00136         {
00137             e.printStackTrace();
00138             return;
00139         }
00140
00141         // Crée et démarre un thread pour envoyer la trame
00142         new Thread()
00143         {
00144             @Override public void run()
00145             {
00146                 byte[] emission = new byte[1024];
00147
00148                 try
00149                 {
00150                     emission = trame.getBytes();
00151                     DatagramPacket paquetRetour = new DatagramPacket(emission, emission.length,
adresseIPDistante, PORT);
00152                     socket.send(paquetRetour);

```

```

00153         Log.d(TAG, "envoyer() : " + trame + " à " + adresseIPDistant + ":" + PORT);
00154     }
00155     catch (IOException e)
00156     {
00157         e.printStackTrace();
00158         Log.d(TAG, "Erreur émission !");
00159     }
00160 }
00161 }.start();
00162 }
00163
00167 public void recevoir()
00168 {
00169     byte[] reception = new byte[1024];
00170
00171     while (socket != null && !socket.isClosed())
00172     {
00173         try
00174         {
00175             final DatagramPacket paquetRecu = new DatagramPacket(reception, reception.length);
00176             socket.receive(paquetRecu);
00177             final String donnees = new String(paquetRecu.getData(), paquetRecu.getOffset(), paquetRecu.
getLength());
00178             Log.d(TAG, "Réception [" + paquetRecu.getAddress().getHostAddress() + ":" + paquetRecu.
getPort() + "] -> " + donnees);
00179
00180             if(verifierTrame(donnees))
00181             {
00182                 // Envoie les données reçues à l'activité
00183                 envoyerMessage(TYPE_RECEPTION, paquetRecu.getAddress().getHostAddress(),
paquetRecu.getPort(), donnees);
00184             }
00185         }
00186         catch (Exception e)
00187         {
00188             e.printStackTrace();
00189             Log.d(TAG, "Erreur réception !");
00190         }
00191     }
00192
00193     Log.d(TAG, "recevoir()");
00194 }
00195
00199 private void envoyerMessage(int type, String adresse, int port, String donnees)
00200 {
00201     if(handler == null)
00202         return;
00203     Message msg = Message.obtain();
00204     msg.what = type;
00205     Bundle b = new Bundle();
00206     b.putString("adresseIP", adresse);
00207     b.putInt("port", port);
00208     b.putString("donnees", donnees);
00209     msg.setData(b);
00210     mutex.lock();
00211     handler.sendMessage(msg);
00212     mutex.unlock();
00213     Log.d(TAG, "envoyerMessage() -> handler.sendMessage()");
00214 }
00215
00221 public String fabriquerTrameDemande(int typeTrame)
00222 {
00223     /*
00224      * Protocole
00225      *
00226      * Demande informations du portier :
00227      * $GET;1\r\n
00228      *
00229      * Demande disponibilité du portier :
00230      * $GET;3\r\n
00231      */
00232
00233     String trame = null;
00234     Log.d(TAG, "fabriquerTrameDemande() type = " + typeTrame);
00235
00236     switch(typeTrame)
00237     {
00238         case DEMANDE_INFORMATIONS:
00239             trame = DELIMITEUR_EN_TETE + "GET;1" + DELIMITEUR_FIN;
00240             break;
00241         case DEMANDE_DISPONIBILITE:
00242             trame = DELIMITEUR_EN_TETE + "GET;3" + DELIMITEUR_FIN;
00243             break;
00244         default:
00245             Log.d(TAG, "fabriquerTrameDemande() : type de trame inconnu !");
00246     }
00247
00248     Log.d(TAG, "fabriquerTrameDemande() trame = " + trame);
00249
00250     return trame;
00251 }

```

```

00252
00258     public String fabriquerTrameModification(int typeTrame, List<String>
parametres)
00259     {
00260         /*
00261          * Protocole
00262          *
00263          * Actualiser les informations d'un portier :
00264          * $SET;1;nomSalle;description;emplacement;surface\r\n
00265          *
00266          * Actualiser la disponibilité d'un portier :
00267          * $SET;3;disponibilité\r\n
00268          *
00269          * Exemple d'utilisation :
00270          * List<String> parametres = Arrays.asList("B11", "Salle de cours", "Batiment BTS", "25");
00271          * String trame =
communication.fabriquerTrameModification(Communication.MODIFICATION_INFORMATIONS, parametres);
00272          */
00273
00274         // Vérifications
00275         if(parametres == null)
00276             return null;
00277
00278         if(parametres.size() < 1)
00279             return null;
00280
00281         Log.d(TAG, "fabriquerTrameModification() type = " + typeTrame);
00282         Log.d(TAG, "fabriquerTrameModification() Nb parametres = " + parametres.size());
00283         for(int i=0;i<parametres.size();++i)
00284         {
00285             Log.d(TAG, "fabriquerTrameModification() parametres[" + i + "] = " + parametres.get(i));
00286         }
00287
00288         String trame = null;
00289
00290         switch(typeTrame)
00291         {
00292             case MODIFICATION_INFORMATIONS:
00293                 if(parametres.size() != NB_CHAMPS_INFORMATIONS)
00294                     return null;
00295                 trame = DELIMITEUR_EN_TETE + "SET;1;" + parametres.get(CHAMP_NOM) + DELIMITEUR_CHAMP +
parametres.get(CHAMP_DESCRIPTION) + DELIMITEUR_CHAMP + parametres.get(CHAMP_LIEU) + DELIMITEUR_CHAMP +
parametres.get(CHAMP_SUPERFICIE) + DELIMITEUR_FIN;
00296                 break;
00297
00298             case MODIFICATION_DISPONIBILITE:
00299                 if(parametres.size() >= NB_CHAMPS_DISPONIBILITE && parametres.size() <= (
NB_CHAMPS_DISPONIBILITE+1))
00300                 {
00301                     if (parametres.get(0).equals("0"))
00302                         trame = DELIMITEUR_EN_TETE + "SET;3;" + parametres.get(0) +
DELIMITEUR_FIN;
00303                     else
00304                         trame = DELIMITEUR_EN_TETE + "SET;3;" + parametres.get(0) + ";" + parametres.get(1)
+ DELIMITEUR_FIN;
00305                 }
00306                 break;
00307
00308             default:
00309                 Log.d(TAG, "fabriquerTrameModification() : type de trame inconnu !");
00310         }
00311
00312         Log.d(TAG, "fabriquerTrameModification() trame = " + trame);
00313
00314         return trame;
00315     }
00316
00321     public boolean verifierTrame(String trame)
00322     {
00323         Log.d(TAG, "verifierTrame() " + (trame.startsWith(DELIMITEUR_EN_TETE) && trame.endsWith(
DELIMITEUR_FIN)));
00324
00325         return (trame.startsWith(DELIMITEUR_EN_TETE) && trame.endsWith(DELIMITEUR_FIN));
00326     }
00327
00331     public void arreter()
00332     {
00333         if(socket == null)
00334             return;
00335         socket.close();
00336     }
00337
00341     @Override
00342     public void run()
00343     {
00344         Log.d(TAG, "Démarre le thread réception");
00345         recevoir();
00346     }
00347
00353     public static int recupererTypeTrame(String[] champs)
00354     {

```

```

00355         int typeTrame = Communication.Trame_INCONNUE;
00356
00357         switch(champs.length)
00358         {
00359             case Communication.NB_CHAMPS_DEMANDE_INFORMATIONS:
00360                 Log.d(TAG, "handleMessage() Trame DEMANDE_INFORMATIONS");
00361                 typeTrame = Communication.DEMANDE_INFORMATIONS;
00362                 break;
00363             case Communication.NB_CHAMPS_DEMANDE_DISPONIBILITE:
00364                 Log.d(TAG, "handleMessage() Trame DEMANDE_DISPONIBILITE");
00365                 typeTrame = Communication.DEMANDE_DISPONIBILITE;
00366                 break;
00367             case Communication.
00368                 NB_CHAMPS_RETOUR_MODIFICATION_DISPONIBILITE:
00369                 Log.d(TAG, "handleMessage() Trame MODIFICATION_DISPONIBILITE");
00370                 typeTrame = Communication.
00371                     MODIFICATION_DISPONIBILITE;
00372         }
00373         return typeTrame;
00374     }

```

9.5 Référence du fichier EspaceDeTravail.java

Déclaration de la classe EspaceDeTravail.

Classes

— class [com.lasalle.meeting.EspaceDeTravail](#)
L'espace de travail.

Paquetages

— package [com.lasalle.meeting](#)

9.5.1 Description détaillée

Déclaration de la classe EspaceDeTravail.

Auteur

KELLER-LAVALLEE Joachim

LastChangedRevision

94

LastChangedDate

2021-06-11 12 :16 :56 +0200 (ven. 11 juin 2021)

Définition dans le fichier [EspaceDeTravail.java](#).

9.6 EspaceDeTravail.java

```

00001 package com.lasalle.meeting;
00002
00003 import android.os.Bundle;
00004 import android.os.Handler;
00005 import android.os.Message;
00006 import android.util.Log;
00007
00008 import org.json.JSONException;
00009 import org.json.JSONObject;
00010
00011 import java.io.Serializable;
00012 import java.util.Arrays;
00013 import java.util.List;
00014
00027 public class EspaceDeTravail implements Serializable
00028 {
00032     private static final String TAG = "_EspaceDeTravail";
00033     public static final int INDICE_CHAUD = 3;
00034     public static final int INDICE_TIEDE = 2;
00035     public static final int INDICE_LEGEREMENT_TIEDE = 1;
00036     public static final int INDICE_NEUTRE = 0;
00037     public static final int INDICE_LEGEREMENT_FRAIS = -1;
00038     public static final int INDICE_FRAIS = -2;
00039     public static final int INDICE_FROID = -3;
00040
00044     private String adresseIP;
00045     private String nom;
00046     private String lieu;
00047     private String description;
00048     private int superficie;
00049     private double temperature;
00050     private int indiceDeConfort;
00051     private boolean estReserve;
00052     private String code;
00053     private boolean estFavori;
00054     private Communication communication = null;
00055
00059     public EspaceDeTravail(String adresseIP)
00060     {
00061         this.adresseIP = adresseIP;
00062         this.nom = "";
00063         this.lieu = "";
00064         this.description = "";
00065         this.superficie = 0;
00066         this.temperature = 0.;
00067         this.indiceDeConfort = 0;
00068         this.estReserve = false;
00069         this.code = "";
00070         this.estFavori = false;
00071
00072         fromJSON(IHMMeeting.recupererDonneesEspaceDeTravail
00073 (this));
00074     }
00079     public String getAdresseIP()
00080     {
00081         return adresseIP;
00082     }
00083
00088     public String getNom()
00089     {
00090         return nom;
00091     }
00092
00097     public String getLieu()
00098     {
00099         return lieu;
00100     }
00101
00106     public String getDescription()
00107     {
00108         return description;
00109     }
00110
00115     public int getSuperficie()
00116     {
00117         return superficie;
00118     }
00119
00124     public double getTemperature()
00125     {
00126         return temperature;
00127     }
00128
00133     public int getIndiceDeConfort()
00134     {
00135         return indiceDeConfort;
00136     }

```

```

00137
00142     public boolean getEstReserve()
00143     {
00144         return estReserve;
00145     }
00146
00151     public String getCode()
00152     {
00153         return code;
00154     }
00155
00160     public boolean getEstFavori()
00161     {
00162         return estFavori;
00163     }
00164
00169     public void setEstReserve(boolean estReserve)
00170     {
00171         this.estReserve = estReserve;
00172     }
00173
00178     public void setCode(String code)
00179     {
00180         this.code = code;
00181         //IHMMeeting.sauvegarderDonneesEspaceDeTravail(this);
00182
00183         Log.d(TAG, "setCode() " + code);
00184     }
00185
00190     public void setEstFavori(boolean estFavori)
00191     {
00192         this.estFavori = estFavori;
00193         IHMMeeting.sauvegarderDonneesEspaceDeTravail(this);
00194
00195         Log.d(TAG, "setEstFavori() " + estFavori);
00196     }
00197
00201     public void reserver()
00202     {
00203         if (communication == null)
00204             return;
00205         Log.d(TAG, "reserver()");
00206
00207         String trame = "\0";
00208         List<String> parametres = Arrays.asList("0");
00209         trame = communication.fabriquerTrameModification(
Communication.MODIFICATION_DISPONIBILITE, parametres);
communication.envoyer(trame, adresseIP);
00210
00211         setEstReserve(true);
00212     }
00213
00214
00218     public void liberer(String code)
00219     {
00220         if (communication == null)
00221             return;
00222         Log.d(TAG, "liberer()");
00223
00224         String trame = "\0";
00225         List<String> parametres = Arrays.asList("1", code);
00226         trame = communication.fabriquerTrameModification(
Communication.MODIFICATION_DISPONIBILITE, parametres);
communication.envoyer(trame, adresseIP);
00227
00228         setEstReserve(false);
00229     }
00230
00231
00236     public void modifierInformations(List<String> parametres)
00237     {
00238         if (communication == null)
00239             return;
00240         Log.d(TAG, "modifierInformations()");
00241
00242         String trame = "\0";
00243         trame = communication.fabriquerTrameModification(
Communication.MODIFICATION_INFORMATIONS, parametres);
communication.envoyer(trame, adresseIP);
00244
00245     }
00246
00251     public boolean extraireInformations(String trame)
00252     {
00253         trame = trame.replace("$", "");
00254         trame = trame.replace("\r\n", "");
00255         String[] champs = trame.split(";");
00256
00257         if (champs.length == Communication.
NB_CHAMPS_DEMANDE_INFORMATIONS)
00258         {
00259             this.nom = champs[Communication.CHAMP_NOM];
00260             this.description = champs[Communication.
CHAMP_DESCRIPTION];

```



```

00266         this.lieu = champs[Communication.CHAMP_LIEU];
00267         if (!champs[Communication.CHAMP_SUPERFICIE].isEmpty())
00268         {
00269             this.superficie = Integer.parseInt(champs[Communication.
CHAMP_SUPERFICIE]);
00270         }
00271         if (!champs[Communication.CHAMP_DISPONIBILITE].isEmpty())
00272         {
00273             if (Integer.parseInt(champs[Communication.
CHAMP_DISPONIBILITE]) == 1)
00274             {
00275                 this.estReserve = false;
00276             }
00277             else
00278             {
00279                 this.estReserve = true;
00280             }
00281         }
00282         if (!champs[Communication.CHAMP_TEMPERATURE].isEmpty())
00283         {
00284             this.temperature = Double.parseDouble(champs[Communication.
CHAMP_TEMPERATURE]);
00285         }
00286         if (!champs[Communication.CHAMP_INDICE_DE_CONFORT].isEmpty())
00287         {
00288             this.indiceDeConfort = Integer.parseInt(champs[Communication.
CHAMP_INDICE_DE_CONFORT]);
00289         }
00290
00291         Log.d(TAG, "extraireInformations() nom : " + nom + " - description : " + description + " - lieu
: " + lieu + " - superficie : " + superficie + " - estReserve : " + estReserve + " - temperature : " +
temperature + " - indiceDeConfort : " + indiceDeConfort);
00292
00293         return true;
00294     }
00295
00296     return false;
00297 }
00298
00303 public boolean extraireCode(String trame)
00304 {
00310     trame = trame.replace("$", "");
00311     trame = trame.replace("\r\n", "");
00312     String[] champs = trame.split(";");
00313
00314     if(champs.length == Communication.
NB_CHAMPS_MODIFICATION_DISPONIBILITE)
00315     {
00316         this.nom = champs[Communication.CHAMP_NOM];
00317
00318         if (!champs[1].isEmpty())
00319         {
00320             this.code = champs[Communication.CHAMP_CODE];
00321         }
00322
00323         Log.d(TAG, "extraireCode() nom : " + nom + " - code : " + code);
00324
00325         return true;
00326     }
00327
00328     return false;
00329 }
00330
00335 public void initialiserCommunication(Handler handler)
00336 {
00337     Log.d(TAG, "initialiserCommunication()");
00338     if(communication == null)
00339     {
00340         communication = new Communication();
00341         communication.setHandler(handler);
00342
00343         // Démarre la réception des trames des portiers
00344         Thread tCommunicationUDP = new Thread(communication, getAdresseIP());
00345         tCommunicationUDP.start(); // execute la méthode run()
00346     }
00347     else
00348     {
00349         communication.arreter();
00350         communication.setHandler(null);
00351         communication = null;
00352     }
00353 }
00354
00358 public void demanderInformations()
00359 {
00360     communication.envoyer(communication.fabriquerTrameDemande(
Communication.DEMANDE_INFORMATIONS), adresseIP);
00361 }
00362
00367 public String toJSON()

```

```

00368     {
00369         JSONObject objet = new JSONObject();
00370         try
00371         {
00372             objet.put("code", this.code);
00373             objet.put("estFavori", this.estFavori);
00374         }
00375         catch (JSONException e)
00376         {
00377             e.printStackTrace();
00378             Log.i(TAG, "toJSON() Erreur !");
00379         }
00380
00381         //Log.i(TAG, "toJSON() JSON = " + objet.toString());
00382         return objet.toString();
00383     }
00384
00389     public void fromJSON(String strJSON)
00390     {
00391         try
00392         {
00393             //Log.i(TAG, "fromJSON() JSON = " + strJSON);
00394             JSONObject json = new JSONObject(strJSON);
00395
00396             this.code = json.getString("code");
00397             this.estFavori = json.getBoolean("estFavori");
00398         }
00399         catch (JSONException e)
00400         {
00401             e.printStackTrace();
00402             Log.i(TAG, "fromJSON() Erreur !");
00403         }
00404     }
00405 }

```

9.7 Référence du fichier EspaceDeTravailAdaptateur.java

Déclaration de la classe EspaceDeTravailAdaptateur.

Classes

- class [com.lasalle.meeting.EspaceDeTravailAdaptateur](#)
L'affichage d'un espace de travail dans la liste des espaces de travail sur la page d'accueil.
- class [com.lasalle.meeting.EspaceDeTravailAdaptateur.ViewHolder](#)

Paquetages

- package [com.lasalle.meeting](#)

9.7.1 Description détaillée

Déclaration de la classe EspaceDeTravailAdaptateur.

Auteur

KELLER-LAVALLEE Joachim

LastChangedRevision

86

LastChangedDate

2021-06-04 17 :29 :36 +0200 (ven. 04 juin 2021)

Définition dans le fichier [EspaceDeTravailAdaptateur.java](#).

9.8 EspaceDeTravailAdaptateur.java

```

00001 package com.lasalle.meeting;
00002
00003 import android.content.Context;
00004 import android.graphics.Color;
00005 import android.util.Log;
00006 import android.view.LayoutInflater;
00007 import android.view.View;
00008 import android.view.ViewGroup;
00009 import android.widget.AdapterView;
00010 import android.widget.AdapterView;
00011 import android.widget.AdapterView;
00012
00013 import java.util.Vector;
00014
00027 public class EspaceDeTravailAdaptateur extends ArrayAdapter<EspaceDeTravail>
00028 {
00029     private static final String TAG = "_EspaceDeTravailAdaptateur";
00030
00031     public EspaceDeTravailAdaptateur(Context context, int resource,
    Vector<EspaceDeTravail> espacesDeTravail)
00032     {
00033         super(context, resource, espacesDeTravail);
00034         Log.d(TAG, "EspaceDeTravailAdaptateur()");
00035     }
00036
00037     private static class ViewHolder
00038     {
00039         TextView nomEspaceDeTravail;
00040         TextView descriptionEspaceDeTravail;
00041         TextView disponibiliteEspaceDeTravail;
00042         TextView indiceDeConfortEspaceDeTravail;
00043         ImageView favoriEspaceDeTravail;
00044     }
00045
00046     @Override
00047     public View getView(int position, View view, ViewGroup parent)
00048     {
00049         EspaceDeTravail espaceDeTravail = null;
00050         ViewHolder viewHolder;
00051
00052         if (view == null)
00053         {
00054             viewHolder = new ViewHolder();
00055             LayoutInflater inflater = LayoutInflater.from(getContext());
00056             view = inflater.inflate(R.layout.element_espace_travail, parent, false);
00057             viewHolder.nomEspaceDeTravail = (TextView) view.findViewById(R.id.nomEspaceDeTravail);
00058             viewHolder.descriptionEspaceDeTravail = (TextView) view.findViewById(R.id.
descriptionEspaceDeTravail);
00059             viewHolder.disponibiliteEspaceDeTravail = (TextView) view.findViewById(R.id.
disponibiliteEspaceDeTravail);
00060             viewHolder.indiceDeConfortEspaceDeTravail = (TextView) view.findViewById(R.id.
indiceDeConfortEspaceDeTravail);
00061             viewHolder.favoriEspaceDeTravail = (ImageView) view.findViewById(R.id.favoriEspaceDeTravail);
00062             view.setTag(viewHolder);
00063         }
00064         else
00065         {
00066             viewHolder = (ViewHolder) view.getTag();
00067         }
00068
00069         espaceDeTravail = getItem(position);
00070         if (espaceDeTravail != null)
00071         {
00072             //Log.d(TAG, "Nom : " + espaceDeTravail.getNom());
00073             viewHolder.nomEspaceDeTravail.setText(espaceDeTravail.getNom());
00074             viewHolder.descriptionEspaceDeTravail.setText(espaceDeTravail.
getDescription());
00075
00076             if (!espaceDeTravail.getEstReserve())
00077             {
00078                 viewHolder.disponibiliteEspaceDeTravail.setText("Libre");
00079                 viewHolder.disponibiliteEspaceDeTravail.setTextColor(Color.parseColor("#00FF00")); //
Color.rgb(0,255,0)
00080             }
00081             else
00082             {
00083                 viewHolder.disponibiliteEspaceDeTravail.setText("Occupé");
00084                 viewHolder.disponibiliteEspaceDeTravail.setTextColor(Color.rgb(255,0,0));
00085             }
00086
00087             switch (espaceDeTravail.getIndiceDeConfort())
00088             {
00089                 case EspaceDeTravail.INDICE_CHAUD:
00090                     viewHolder.indiceDeConfortEspaceDeTravail.setText("Chaud");
00091                     break;
00092
00093                 case EspaceDeTravail.INDICE_TIEDE:
00094                     viewHolder.indiceDeConfortEspaceDeTravail.setText("Tiède");

```

```

00095         break;
00096
00097     case EspaceDeTravail.INDICE_LEGEREMENT_TIEDE:
00098         viewHolder.indiceDeConfortEspaceDeTravail.setText("Légèrement tiède");
00099         break;
00100
00101     case EspaceDeTravail.INDICE_NEUTRE:
00102         viewHolder.indiceDeConfortEspaceDeTravail.setText("Neutre");
00103         break;
00104
00105     case EspaceDeTravail.INDICE_LEGEREMENT_FRAIS:
00106         viewHolder.indiceDeConfortEspaceDeTravail.setText("Légèrement frais");
00107         break;
00108
00109     case EspaceDeTravail.INDICE_FRAIS:
00110         viewHolder.indiceDeConfortEspaceDeTravail.setText("Frais");
00111         break;
00112
00113     case EspaceDeTravail.INDICE_FROID:
00114         viewHolder.indiceDeConfortEspaceDeTravail.setText("Froid");
00115         break;
00116     }
00117
00118     if(!espaceDeTravail.getEstFavori())
00119     {
00120         viewHolder.favoriEspaceDeTravail.setVisibility(View.INVISIBLE);
00121     }
00122     else
00123     {
00124         viewHolder.favoriEspaceDeTravail.setVisibility(View.VISIBLE);
00125     }
00126 }
00127
00128 return view;
00129 }
00130 }

```

9.9 Référence du fichier IHMMeeting.java

Déclaration de la classe IHMMeeting.

Classes

- class [com.lasalle.meeting.IHMMeeting](#)
L'activité principale de l'application Meeting.

Paquetages

- package [com.lasalle.meeting](#)

9.9.1 Description détaillée

Déclaration de la classe IHMMeeting.

Auteur

KELLER-LAVALLEE Joachim

LastChangedRevision

94

LastChangedDate

2021-06-11 12 :16 :56 +0200 (ven. 11 juin 2021)

Définition dans le fichier [IHMMeeting.java](#).

9.10 IHMMeeting.java

```

00001 package com.lasalle.meeting;
00002
00003 import androidx.appcompat.app.AlertDialog;
00004 import androidx.appcompat.app.AppCompatActivity;
00005 import androidx.swiperefreshlayout.widget.SwipeRefreshLayout;
00006
00007 import android.content.Context;
00008 import android.content.DialogInterface;
00009 import android.content.Intent;
00010 import android.content.SharedPreferences;
00011 import android.net.wifi.WifiManager;
00012 import android.os.Bundle;
00013 import android.os.Handler;
00014 import android.os.Message;
00015 import android.util.Log;
00016 import android.view.LayoutInflater;
00017 import android.view.Menu;
00018 import android.view.MenuItem;
00019 import android.view.View;
00020 import android.widget.AdapterView;
00021 import android.widget.EditText;
00022 import android.widget.ListView;
00023 import android.widget.RadioButton;
00024 import android.widget.RadioGroup;
00025 import android.widget.TextView;
00026
00027 import java.io.Serializable;
00028 import java.util.Collections;
00029 import java.util.Comparator;
00030 import java.util.Vector;
00031
00044 public class IHMMeeting extends AppCompatActivity
00045 {
00049     private static final String TAG = "_IHMMeeting";
00050     private static final String PREFERENCES = "preferences";
00051     private static final String PREFERENCES_CODE = "code";
00052     private static final String PREFERENCES_EST_FAVORI = "false";
00053
00057     private SwipeRefreshLayout swipeRefreshLayout;
00058     private ListView listeEspacesDeTravail;
00059     private TextView titreEspacesDeTravail;
00060     private AlertDialog.Builder boiteDeDialogueAPropos;
00061     private AlertDialog boiteDeDialogueRechercher;
00062     private AlertDialog.Builder boiteDeDialogueFiltrerParDisponibilite
00063 ;
00064     private AlertDialog.Builder boiteDeDialogueFiltrerParNiveauDeConfort
00065 ;
00066     private String motCle;
00067     private boolean estReserve; //<! Disponibilité sélectionnée pour filtrer la liste des espaces
de travail
00068     private int indiceDeConfort; //<! Niveau de confort sélectionné pour filtrer la liste
des espaces de travail
00069
00071     private static Vector<EspaceDeTravail> espacesDeTravail;
00072     private static Vector<EspaceDeTravail> espacesDeTravailFiltres;
00073     private EspaceDeTravailAdaptateur adaptateur;
00074     private static Communication communication = null;
00075     private WifiManager wm = null;
00076     private int choixFiltre = -1;
00077     private static SharedPreferences preferences;
00078
00082     @Override
00083     protected void onCreate(Bundle savedInstanceState)
00084     {
00085         super.onCreate(savedInstanceState);
00086         setContentView(R.layout.activity_main);
00087         Log.d(TAG, "onCreate()");
00088
00089         initialiserPreferences();
00090
00091         swipeRefreshLayout = (SwipeRefreshLayout) findViewById(R.id.swipeRefreshLayout);
00092         swipeRefreshLayout.setOnRefreshListener(new SwipeRefreshLayout.OnRefreshListener()
00093         {
00094             @Override
00095             public void onRefresh()
00096             {
00097                 Log.d(TAG, "Pull To Refresh");
00098                 initialiserEspacesDeTravail();
00099             }
00100         });
00101
00102         initialiserListeEspacesDeTravail();
00103
00104         afficherListeEspacesDeTravail();
00105
00106         demarrerReseau();
00107
00108         initialiserBoitesDeDialogue();

```

```

00109     }
00110
00114     private void initialiserPreferences()
00115     {
00116         preferences = getBaseContext().getSharedPreferences(PREFERENCES, MODE_PRIVATE);
00117     }
00118
00122     private void demarrerReseau()
00123     {
00124         wm = (WifiManager) getApplicationContext().getSystemService(Context.WIFI_SERVICE);
00125         if (!wm.isWifiEnabled())
00126         {
00127             Log.d(TAG, "WiFi indisponible !");
00128             wm.setWifiEnabled(true);
00129         }
00130         else
00131         {
00132             Log.d(TAG, "WiFi disponible");
00133         }
00134
00135         communication = new Communication(handler);
00136
00137         // Démarre la réception des trames des portiers
00138         Thread tCommunicationUDP = new Thread(communication, "Communication");
00139         tCommunicationUDP.start(); // execute la méthode run()
00140
00141         if(communication != null)
00142         {
00143             // Demande les informations des portiers joignables
00144             communication.envoyer(communication.fabriquerTrameDemande(
Communication.DEMANDE_INFORMATIONS),
Communication.adresseMulticast);
00145         }
00146     }
00147
00151     @Override
00152     protected void onStart()
00153     {
00154         super.onStart();
00155         Log.d(TAG, "onStart()");
00156
00157         initialiserEspacesDeTravail();
00158     }
00159
00163     @Override
00164     protected void onResume()
00165     {
00166         super.onResume();
00167         Log.d(TAG, "onResume()");
00168     }
00169
00173     @Override
00174     protected void onPause()
00175     {
00176         super.onPause();
00177         Log.d(TAG, "onPause()");
00178     }
00179
00183     @Override
00184     protected void onStop()
00185     {
00186         super.onStop();
00187         Log.d(TAG, "onStop()");
00188     }
00189
00193     @Override
00194     protected void onDestroy()
00195     {
00196         super.onDestroy();
00197         Log.d(TAG, "onDestroy()");
00198     }
00199
00203     private void initialiserEspacesDeTravail()
00204     {
00205         Log.d(TAG, "initialiserEspacesDeTravail()");
00206
00207         espacesDeTravail.clear();
00208
00209         if(communication != null)
00210         {
00211             // Demande les informations des portiers joignables
00212             communication.envoyer(communication.fabriquerTrameDemande(
Communication.DEMANDE_INFORMATIONS),
Communication.adresseMulticast);
00213         }
00214
00215         swipeRefreshLayout.setRefreshing(false); // arrête le Pull To Refresh
00216     }
00217
00222     private void ajouterEspaceDeTravail(EspaceDeTravail
espaceDeTravail)

```

```

00223     {
00224         int position = verifierPresenceEspaceDeTravail(espaceDeTravail);
00225
00226         if(position == -1)
00227         {
00228             espacesDeTravail.add(espaceDeTravail);
00229             rafraichirListeEspacesFiltres();
00230         }
00231     }
00232
00237     private void modifierEspaceDeTravail(EspaceDeTravail
espaceDeTravail)
00238     {
00239         int position = verifierPresenceEspaceDeTravail(espaceDeTravail);
00240         Log.d(TAG, "modifierEspaceDeTravail() : position = " + position);
00241
00242         if(position != -1)
00243         {
00244             //espacesDeTravail.removeElementAt(position);
00245             //espacesDeTravail.add(espaceDeTravail);
00246             espacesDeTravail.set(position, espaceDeTravail);
00247             rafraichirListeEspacesFiltres();
00248         }
00249     }
00250
00255     private void supprimerEspaceDeTravail(
EspaceDeTravail espaceDeTravail)
00256     {
00257         int position = verifierPresenceEspaceDeTravail(espaceDeTravail);
00258
00259         if(position != -1)
00260         {
00261             espacesDeTravail.removeElementAt(position);
00262             rafraichirListeEspacesFiltres();
00263         }
00264     }
00265
00271     private int verifierPresenceEspaceDeTravail(
EspaceDeTravail espaceDeTravail)
00272     {
00273         for(int i = 0; i < espacesDeTravail.size(); ++i)
00274         {
00275             if(espaceDeTravail.getNom().equals(espacesDeTravail.elementAt(i).getNom()))
00276             {
00277                 return i;
00278             }
00279         }
00280         return -1;
00281     }
00282
00288     private int verifierPresenceEspaceDeTravail(String adresseIP)
00289     {
00290         for(int i = 0; i < espacesDeTravail.size(); ++i)
00291         {
00292             if(espacesDeTravail.elementAt(i).getAdresseIP().equals(adresseIP))
00293             {
00294                 return i;
00295             }
00296         }
00297         return -1;
00298     }
00299
00303     private void initialiserListeEspacesDeTravail()
00304     {
00305         Log.d(TAG, "initialiserListeEspacesDeTravail()");
00306
00307         espacesDeTravail = new Vector<EspaceDeTravail>();
00308         espacesDeTravailFiltres = new Vector<EspaceDeTravail>();
00309         choixFiltre = R.id.actionAfficherTous;
00310
00311         listeEspacesDeTravail = (ListView) findViewById(R.id.listeEspacesDeTravail);
00312         titreEspacesDeTravail = (TextView) findViewById(R.id.titreEspacesDeTravail);
00313         titreEspacesDeTravail.setText("Tous les espaces de travail détectés");
00314     }
00315
00319     private void afficherListeEspacesDeTravail()
00320     {
00321         Log.d(TAG, "afficherListeEspacesDeTravail()");
00322
00323         adaptateur = new EspaceDeTravailAdaptateur(this, R.layout.
element_espace_travail, espacesDeTravailFiltres);
00324
00325         listeEspacesDeTravail.setAdapter(adaptateur);
00326         adaptateur.setNotifyOnChange(true);
00327
00328         listeEspacesDeTravail.setOnItemClickListener(
00329             new AdapterView.OnItemClickListener()
00330             {
00331                 @Override
00332                 public void onItemClick(AdapterView<?> a, View v, int position, long id)
00333             {

```

```

00334         Log.d(TAG, "Position : " + position + " - " + " Nom : " + espacesDeTravailFiltres.
get(position).getNom());
00335         Intent intent = new Intent(IHMMeeting.this,
AffichageEspaceDeTravail.class);
00336         intent.putExtra("unEspaceDeTravail", (Serializable)espacesDeTravailFiltres.get(
position));
00337         startActivityForResult(intent, 0);
00338     }
00339 }
00340 );
00341 }
00342
00346 @Override
00347 protected void onActivityResult(int requestCode, int resultCode, Intent data)
00348 {
00349     super.onActivityResult(requestCode, resultCode, data);
00350     Log.d(TAG, "onActivityResult() requestCode=" + requestCode + " - resultCode=" + resultCode + "");
00351     /*EspaceDeTravail espaceDeTravail =
(EspaceDeTravail)data.getSerializableExtra("unEspaceDeTravail");
00352     Log.d(TAG, "onActivityResult() espaceDeTravail : " + espaceDeTravail.getNom() + " - " +
espaceDeTravail.getDescription() + " - " + espaceDeTravail.getLieu() + " - " + espaceDeTravail.getSuperficie() + " - "
+ espaceDeTravail.getEstReserve());
00353     modifierEspaceDeTravail(espaceDeTravail);*/
00354 }
00355
00360 private Handler handler = new Handler()
00361 {
00362     @Override
00363     public void handleMessage(Message msg)
00364     {
00365         super.handleMessage(msg);
00366         Bundle b = msg.getData();
00367
00368         switch(msg.what)
00369         {
00370             case Communication.TYPE_RECEPTION:
00371                 String trame = b.getString("donnees");
00372                 Log.d(TAG, "handleMessage() Réception [" + b.getString("adresseIP") + ":" + b.getInt("
port") + "] -> " + trame);
00373
00374                 String[] champs = trame.split(";");
00375                 int typeTrame = Communication.
recupererTypeTrame(champs);
00376
00377                 switch(typeTrame)
00378                 {
00379                     case Communication.DEMANDE_INFORMATIONS:
00380                         // un nouvel espace de travail ?
00381                         int numeroEspaceDeTravail =
verifierPresenceEspaceDeTravail(b.getString("adresseIP"));
00382                         if(numeroEspaceDeTravail == -1)
00383                         {
00384                             EspaceDeTravail espaceDeTravail = new
EspaceDeTravail(b.getString("adresseIP"));
00385                             espaceDeTravail.extraireInformations(trame);
00386                             ajouterEspaceDeTravail(espaceDeTravail);
00387                         }
00388                     else
00389                     {
00390                         espacesDeTravail.get(numeroEspaceDeTravail).extraireInformations(trame);
00391                     }
00392                     break;
00393
00394                     case Communication.
MODIFICATION_DISPONIBILITE:
00395                         numeroEspaceDeTravail =
verifierPresenceEspaceDeTravail(b.getString("adresseIP"));
00396                         espacesDeTravail.get(numeroEspaceDeTravail).extraireCode(trame);
00397                         break;
00398
00399                     default:
00400                         Log.d(TAG, "handleMessage() : type de trame inconnu !");
00401                 }
00402                 break;
00403
00404                 default:
00405                     Log.d(TAG, "handleMessage() : code inconnu !");
00406             }
00407         }
00408     };
00409
00413 @Override
00414 public boolean onCreateOptionsMenu(Menu menu)
00415 {
00416     getMenuInflater().inflate(R.menu.menu_main, menu);
00417     return true;
00418 }
00419
00423 @Override
00424 public boolean onOptionsItemSelected(MenuItem item)
00425 {

```



```

00426         int id = item.getItemId();
00427
00428         switch(id)
00429         {
00430             case R.id.actionAfficherTous:
00431                 Log.d(TAG, "onOptionsItemSelected() actionAfficherTous");
00432                 choixFiltre = R.id.actionAfficherTous;
00433                 titreEspacesDeTravail.setText("Tous les espaces de travail détectés");
00434                 filtrerTous();
00435                 break;
00436
00437             case R.id.actionAfficherFavoris:
00438                 Log.d(TAG, "onOptionsItemSelected() actionAfficherFavoris");
00439                 choixFiltre = R.id.actionAfficherFavoris;
00440                 titreEspacesDeTravail.setText("Favoris");
00441                 filtrerParFavoris();
00442                 break;
00443
00444             case R.id.actionRechercher:
00445                 Log.d(TAG, "onOptionsItemSelected() actionRechercher");
00446                 choixFiltre = R.id.actionRechercher;
00447                 titreEspacesDeTravail.setText("Recherche");
00448                 boiteDeDialogueRechercher.show();
00449                 EditText saisieMotCle = (EditText) ((AlertDialog)
00450                     boiteDeDialogueRechercher).findViewById(R.id.saisieMotCle);
00451                 saisieMotCle.setText(motCle);
00452                 break;
00453
00454             case R.id.actionFiltrerParDisponibilite:
00455                 Log.d(TAG, "onOptionsItemSelected() actionFiltrerParDisponibilite");
00456                 choixFiltre = R.id.actionFiltrerParDisponibilite;
00457                 titreEspacesDeTravail.setText("Espaces de travail filtrés par disponibilité");
00458                 boiteDeDialogueFiltrerParDisponibilite.show();
00459                 break;
00460
00461             case R.id.actionFiltrerParNiveauDeConfort:
00462                 Log.d(TAG, "onOptionsItemSelected() actionFiltrerParNiveauDeConfort");
00463                 choixFiltre = R.id.actionFiltrerParNiveauDeConfort;
00464                 titreEspacesDeTravail.setText("Espaces de travail filtrés par niveau de confort");
00465                 boiteDeDialogueFiltrerParNiveauDeConfort.show();
00466                 break;
00467
00468             case R.id.actionAPropos:
00469                 Log.d(TAG, "onOptionsItemSelected() actionAPropos");
00470                 boiteDeDialogueAPropos.show();
00471                 break;
00472         }
00473         return super.onOptionsItemSelected(item);
00474     }
00475
00479     public void initialiserBoitesDeDialogue()
00480     {
00481         boiteDeDialogueAPropos = new AlertDialog.Builder(this);
00482         boiteDeDialogueAPropos.setTitle("À propos");
00483         String message = "Projet Meeting version " + BuildConfig.VERSION_NAME + "\n\nAuteur :
00484         KELLER-LAVALLEE Joachim\nBTS SNIR La Salle Avignon 2021";
00485         boiteDeDialogueAPropos.setMessage(message);
00486
00487         initialiserBoiteDeDialogueRechercher();
00488         initialiserBoiteDeDialogueFiltrerParDisponibilite();
00489         initialiserBoiteDeDialogueFiltrerParNiveauDeConfort();
00490     }
00494     private void initialiserBoiteDeDialogueRechercher()
00495     {
00496         AlertDialog.Builder boiteDeDialogueRechercher = new AlertDialog.Builder(this);
00497         boiteDeDialogueRechercher.setTitle("Rechercher un espace de travail");
00498         LayoutInflater inflater = this.getLayoutInflater();
00499         View vue = inflater.inflate(R.layout.boite_recherche, null);
00500         boiteDeDialogueRechercher.setView(vue);
00501
00502         boiteDeDialogueRechercher.setPositiveButton("Rechercher", new DialogInterface.OnClickListener()
00503         {
00504             public void onClick(DialogInterface dialog, int which)
00505             {
00506                 EditText saisieMotCle = (EditText) ((AlertDialog) dialog).findViewById(R.id.saisieMotCle);
00507                 motCle = saisieMotCle.getText().toString();
00508                 titreEspacesDeTravail.setText("Résultats de la recherche : " + motCle);
00509                 rechercher(motCle);
00510             }
00511         });
00512         boiteDeDialogueRechercher.setNegativeButton("Annuler", new DialogInterface.OnClickListener()
00513         {
00514             public void onClick(DialogInterface dialog, int which)
00515             {
00516             }
00517         });
00518     }

```

```

00519
00520         this.boiteDeDialogueRechercher = boiteDeDialogueRechercher.create();
00521     }
00522
00526     private void initialiserBoiteDeDialogueFiltrerParDisponibilite
00527     ()
00528     {
00529         boiteDeDialogueFiltrerParDisponibilite = new AlertDialog.
00530         Builder(this);
00531         boiteDeDialogueFiltrerParDisponibilite.setTitle("Filtrer par
00532         disponibilité");
00533         LayoutInflater inflater = this.getLayoutInflater();
00534         View vue = inflater.inflate(R.layout.boite_filtre_disponibilite, null);
00535         boiteDeDialogueFiltrerParDisponibilite.setView(vue);
00536
00537         boiteDeDialogueFiltrerParDisponibilite.setPositiveButton("
00538         Filtrer", new DialogInterface.OnClickListener()
00539         {
00540             public void onClick(DialogInterface dialog, int which)
00541             {
00542                 if(estReserve)
00543                 {
00544                     titreEspacesDeTravail.setText("Espaces de travail occupés");
00545                 }
00546                 else
00547                 {
00548                     titreEspacesDeTravail.setText("Espaces de travail libres");
00549                 }
00550                 filtrerParDisponibilite(estReserve);
00551             }
00552         });
00553         boiteDeDialogueFiltrerParDisponibilite.setNegativeButton("
00554         Annuler", new DialogInterface.OnClickListener()
00555         {
00556             public void onClick(DialogInterface dialog, int which)
00557             {
00558                 //
00559             }
00560         });
00561     }
00562
00563     private void initialiserBoiteDeDialogueFiltrerParNiveauDeConfort
00564     ()
00565     {
00566         boiteDeDialogueFiltrerParNiveauDeConfort = new AlertDialog.
00567         Builder(this);
00568         boiteDeDialogueFiltrerParNiveauDeConfort.setTitle("Filtrer
00569         par niveau de confort");
00570         LayoutInflater inflater = this.getLayoutInflater();
00571         View vue = inflater.inflate(R.layout.boite_filtre_niveau_de_confort, null);
00572         boiteDeDialogueFiltrerParNiveauDeConfort.setView(vue);
00573
00574         boiteDeDialogueFiltrerParNiveauDeConfort.setPositiveButton(
00575         "Filtrer", new DialogInterface.OnClickListener()
00576         {
00577             public void onClick(DialogInterface dialog, int which)
00578             {
00579                 switch(indiceDeConfort)
00580                 {
00581                     case EspaceDeTravail.INDICE_CHAUD:
00582                         titreEspacesDeTravail.setText("Espaces de travail dont le niveau de confort est
00583                         Chaud");
00584                         break;
00585                     case EspaceDeTravail.INDICE_TIEDE:
00586                         titreEspacesDeTravail.setText("Espaces de travail dont le niveau de confort est
00587                         Tiède");
00588                         break;
00589                     case EspaceDeTravail.INDICE_LEGEREMENT_TIEDE:
00590                         titreEspacesDeTravail.setText("Espaces de travail dont le niveau de confort est
00591                         Légèrement tiède");
00592                         break;
00593                     case EspaceDeTravail.INDICE_NEUTRE:
00594                         titreEspacesDeTravail.setText("Espaces de travail dont le niveau de confort est
00595                         Neutre");
00596                         break;
00597                     case EspaceDeTravail.INDICE_LEGEREMENT_FRAIS:
00598                         titreEspacesDeTravail.setText("Espaces de travail dont le niveau de confort est
00599                         Légèrement frais");
00600                         break;
00601                     case EspaceDeTravail.INDICE_FRAIS:
00602                         titreEspacesDeTravail.setText("Espaces de travail dont le niveau de confort est
00603                         Frais");
00604                         break;
00605                     case EspaceDeTravail.INDICE_FROID:
00606                         titreEspacesDeTravail.setText("Espaces de travail dont le niveau de confort est
00607                         Froid");
00608                         break;
00609                 }
00610             }
00611         });
00612     }

```

```

00601         titreEspacesDeTravail.setText("Espaces de travail dont le niveau de confort est
00602         Froid");
00603         break;
00604     }
00605     filtrerParNiveauDeConfort(indiceDeConfort);
00606 }
00607 });
00608 boiteDeDialogueFiltrerParNiveauDeConfort.setNegativeButton(
00609     "Annuler", new DialogInterface.OnClickListener()
00610     {
00611         public void onClick(DialogInterface dialog, int which)
00612         {
00613         }
00614     });
00615 }
00616
00620 public void onRadioButtonClicked(View vue)
00621 {
00622     boolean estCoche = ((RadioButton) vue).isChecked();
00623
00624     switch (vue.getId())
00625     {
00626         case R.id.boutonRadioLibre:
00627             if (estCoche)
00628             {
00629                 estReserve = false;
00630             }
00631             break;
00632
00633         case R.id.boutonRadioOccupe:
00634             if (estCoche)
00635             {
00636                 estReserve = true;
00637             }
00638             break;
00639
00640         case R.id.boutonRadioChaud:
00641             if (estCoche)
00642             {
00643                 indiceDeConfort = EspaceDeTravail.
00644                 INDICE_CHAUD;
00645             }
00646             break;
00647
00648         case R.id.boutonRadioTiede:
00649             if (estCoche)
00650             {
00651                 indiceDeConfort = EspaceDeTravail.
00652                 INDICE_TIEDE;
00653             }
00654             break;
00655
00656         case R.id.boutonRadioLegerementTiede:
00657             if (estCoche)
00658             {
00659                 indiceDeConfort = EspaceDeTravail.
00660                 INDICE_LEGEREMENT_TIEDE;
00661             }
00662             break;
00663
00664         case R.id.boutonRadioNeutre:
00665             if (estCoche)
00666             {
00667                 indiceDeConfort = EspaceDeTravail.
00668                 INDICE_NEUTRE;
00669             }
00670             break;
00671
00672         case R.id.boutonRadioLegerementFrais:
00673             if (estCoche)
00674             {
00675                 indiceDeConfort = EspaceDeTravail.
00676                 INDICE_LEGEREMENT_FRAIS;
00677             }
00678             break;
00679
00680         case R.id.boutonRadioFrais:
00681             if (estCoche)
00682             {
00683                 indiceDeConfort = EspaceDeTravail.
00684                 INDICE_FRAIS;
00685             }
00686             break;
00687
00688         case R.id.boutonRadioFroid:
00689             if (estCoche)
00690             {
00691                 indiceDeConfort = EspaceDeTravail.
00692                 INDICE_FROID;
00693             }
00694             break;
00695     }
00696 }

```

```

00686         }
00687         break;
00688     }
00689 }
00690
00694 private void rechercher(String motCle)
00695 {
00696     espacesDeTravailFiltres.clear();
00697
00698     for(int i = 0; i < espacesDeTravail.size(); i++)
00699     {
00700         if( containsIgnoreCase(espacesDeTravail.elementAt(i).getNom(),
motCle) || containsIgnoreCase(espacesDeTravail.elementAt(i).getDescription(),
motCle) || containsIgnoreCase(espacesDeTravail.elementAt(i).getLieu(),
motCle) || containsIgnoreCase(espacesDeTravail.elementAt(i).getAdresseIP(),
motCle) )
        {
00701             espacesDeTravailFiltres.add(espacesDeTravail.elementAt(i));
00702         }
00703     }
00704
00705     Log.d(TAG, "rechercher() " + motCle);
00706
00707     rafraichirAffichageListeEspaces();
00708 }
00709
00710
00714 private void filtrerParDisponibilite(boolean disponibilite)
00715 {
00716     espacesDeTravailFiltres.clear();
00717
00718     for(int i = 0; i < espacesDeTravail.size(); i++)
00719     {
00720         if(espacesDeTravail.elementAt(i).getEstReserve() == disponibilite)
00721         {
00722             espacesDeTravailFiltres.add(espacesDeTravail.elementAt(i));
00723         }
00724     }
00725
00726     Log.d(TAG, "filtrerParDisponibilite() " + disponibilite);
00727
00728     rafraichirAffichageListeEspaces();
00729 }
00730
00734 private void filtrerParNiveauDeConfort(int indiceDeConfort)
00735 {
00736     espacesDeTravailFiltres.clear();
00737
00738     for(int i = 0; i < espacesDeTravail.size(); i++)
00739     {
00740         if(espacesDeTravail.elementAt(i).getIndiceDeConfort() ==
indiceDeConfort)
        {
00741             espacesDeTravailFiltres.add(espacesDeTravail.elementAt(i));
00742         }
00743     }
00744
00745     Log.d(TAG, "filtrerParNiveauDeConfort() " + indiceDeConfort);
00746
00747     rafraichirAffichageListeEspaces();
00748 }
00749
00750
00754 private void filtrerParFavoris()
00755 {
00756     espacesDeTravailFiltres.clear();
00757
00758     for(int i = 0; i < espacesDeTravail.size(); i++)
00759     {
00760         if(espacesDeTravail.elementAt(i).getEstFavori())
00761         {
00762             espacesDeTravailFiltres.add(espacesDeTravail.elementAt(i));
00763         }
00764     }
00765
00766     Log.d(TAG, "filtrerParFavoris()");
00767
00768     rafraichirAffichageListeEspaces();
00769 }
00770
00774 private void filtrerTous()
00775 {
00776     espacesDeTravailFiltres.clear();
00777
00778     for(int i = 0; i < espacesDeTravail.size(); i++)
00779     {
00780         espacesDeTravailFiltres.add(espacesDeTravail.elementAt(i));
00781     }
00782
00783     Log.d(TAG, "filtrerTous");
00784
00785     rafraichirAffichageListeEspaces();
00786 }

```

```

00787
00791     private void rafraichirListeEspacesFiltres()
00792     {
00793         switch(choixFiltre)
00794         {
00795             case R.id.actionAfficherTous:
00796                 titreEspacesDeTravail.setText("Tous les espaces de travail détectés");
00797                 filtrerTous();
00798                 break;
00799
00800             case R.id.actionAfficherFavoris:
00801                 titreEspacesDeTravail.setText("Favoris");
00802                 filtrerParFavoris();
00803                 break;
00804
00805             case R.id.actionFiltrerParDisponibilite:
00806                 titreEspacesDeTravail.setText("Espaces de travail filtrés par disponibilité");
00807                 filtrerParDisponibilite(estReserve);
00808                 break;
00809
00810             case R.id.actionFiltrerParNiveauDeConfort:
00811                 titreEspacesDeTravail.setText("Espaces de travail filtrés par niveau de confort");
00812                 filtrerParNiveauDeConfort(indiceDeConfort);
00813                 break;
00814         }
00815     }
00816
00820     private void rafraichirAffichageListeEspaces()
00821     {
00822         trierEspacesDeTravail("nom", espacesDeTravailFiltres);
00823         swipeRefreshLayout.setRefreshing(false);
00824         adaptateur.notifyDataSetChanged();
00825     }
00826
00831     private void trierEspacesDeTravail(final String champ, Vector<EspaceDeTravail>
lesEspacesDeTravail)
00832     {
00833         //Log.d(TAG, "trierEspacesDeTravail() champ = " + champ + " - nb = " + lesEspacesDeTravail.size());
00834         Collections.sort(lesEspacesDeTravail, new Comparator<EspaceDeTravail>()
00835         {
00836             @Override
00837             public int compare(EspaceDeTravail e1,
EspaceDeTravail e2)
00838             {
00839                 if(champ.equals("nom"))
00840                 {
00841                     return e1.getNom().compareTo(e2.getNom());
00842                 }
00843                 else if(champ.equals("adresseIP"))
00844                 {
00845                     return e1.getAdresseIP().compareTo(e2.
getAdresseIP());
00846                 }
00847                 else if(champ.equals("superficie"))
00848                 {
00849                     return (e1.getSuperficie() - e2.getSuperficie());
00850                 }
00851                 return e1.getNom().compareTo(e2.getNom());
00852             }
00853         });
00854     }
00855
00859     public static String recupererDonneesEspaceDeTravail(
EspaceDeTravail espaceDeTravail)
00860     {
00861         String donnees = "";
00862
00863         espaceDeTravail.setCode(preferences.getString(PREFERENCES_CODE, null));
00864         espaceDeTravail.setEstFavori(preferences.getBoolean(PREFERENCES_EST_FAVORI, false));
00865
00866         donnees = espaceDeTravail.toJSON();
00867
00868         return donnees;
00869     }
00870
00874     public static void sauvegarderDonneesEspaceDeTravail(
EspaceDeTravail espaceDeTravail)
00875     {
00876         preferences.edit().putString(PREFERENCES_CODE, espaceDeTravail.getCode()).apply();
00877         preferences.edit().putBoolean(PREFERENCES_EST_FAVORI, espaceDeTravail.
getEstFavori()).apply();
00878     }
00879
00884     public static boolean containsIgnoreCase(String str, String searchStr)
00885     {
00886         if(str == null || searchStr == null) return false;
00887
00888         final int length = searchStr.length();
00889         if (length == 0)
00890             return true;
00891

```

```

00892         for (int i = str.length() - length; i >= 0; i--)
00893         {
00894             if (str.regionMatches(true, i, searchStr, 0, length))
00895                 return true;
00896         }
00897         return false;
00898     }
00899 }

```

9.11 Référence du fichier ModificationEspaceDeTravail.java

Déclaration de la classe ModificationEspaceDeTravail.

Classes

- class [com.lasalle.meeting.ModificationEspaceDeTravail](#)
L'activité de modification d'un espace de travail de l'application Meeting.

Paquetages

- package [com.lasalle.meeting](#)

9.11.1 Description détaillée

Déclaration de la classe ModificationEspaceDeTravail.

Auteur

KELLER-LAVALLEE Joachim

Définition dans le fichier [ModificationEspaceDeTravail.java](#).

9.12 ModificationEspaceDeTravail.java

```

00001 package com.lasalle.meeting;
00002
00003 import androidx.appcompat.app.AppCompatActivity;
00004
00005 import android.content.Intent;
00006 import android.os.Bundle;
00007 import android.os.Handler;
00008 import android.os.Message;
00009 import android.util.Log;
00010 import android.view.View;
00011 import android.widget.Button;
00012 import android.widget.EditText;
00013 import android.widget.TextView;
00014
00015 import com.google.android.material.textfield.TextInputEditText;
00016
00017 import java.util.Arrays;
00018 import java.util.List;
00019
00030 public class ModificationEspaceDeTravail extends AppCompatActivity
00031 {
00035     private static final String TAG = "_ModificationEspaceDeTravail";
00036
00040     EditText editionNom;
00041     EditText editionLieu;
00042     EditText editionDescription;
00043     EditText editionSuperficie;
00044
00048     private EspaceDeTravail espaceDeTravail;
00049
00053     @Override
00054     protected void onCreate(Bundle savedInstanceState)

```

```

00055     {
00056         super.onCreate(savedInstanceState);
00057         setContentView(R.layout.activity_modification_espace_de_travail);
00058         Intent intent = getIntent();
00059         espaceDeTravail = (EspaceDeTravail) intent.getSerializableExtra("unEspaceDeTravail");
00060         espaceDeTravail.initialiserCommunication(handler);
00061
00062         afficherEditionNom();
00063         afficherEditionLieu();
00064         afficherEditionDescription();
00065         afficherEditionSuperficie();
00066         afficherBoutons();
00067     }
00068
00072     public void afficherEditionNom()
00073     {
00074         editionNom = (EditText) findViewById(R.id.editionNom);
00075         editionNom.setText(espaceDeTravail.getNom());
00076
00077         Log.d(TAG, "afficherEditionNom() " + espaceDeTravail.getNom());
00078     }
00082     public void afficherEditionLieu()
00083     {
00084         editionLieu = (EditText) findViewById(R.id.editionLieu);
00085         editionLieu.setText(espaceDeTravail.getLieu());
00086
00087         Log.d(TAG, "afficherEditionLieu() " + espaceDeTravail.getLieu());
00088     }
00093     public void afficherEditionDescription()
00094     {
00095         editionDescription = (EditText) findViewById(R.id.editionDescription);
00096         editionDescription.setText(espaceDeTravail.getDescription());
00097
00098         Log.d(TAG, "afficherEditionDescription() " + espaceDeTravail.
00099         getDescription());
00100     }
00104     public void afficherEditionSuperficie()
00105     {
00106         editionSuperficie = (EditText) findViewById(R.id.editionSuperficie);
00107         int superficie = espaceDeTravail.getSuperficie();
00108         editionSuperficie.setText(String.valueOf(superficie));
00109
00110         Log.d(TAG, "afficherEditionSuperficie() " + espaceDeTravail.
00111         getSuperficie());
00112     }
00116     public void afficherBoutons()
00117     {
00118         Button boutonEnregistrer = (Button) findViewById(R.id.boutonEnregistrer);
00119
00120         boutonEnregistrer.setOnClickListener(
00121             new View.OnClickListener()
00122             {
00123                 public void onClick(View v)
00124                 {
00125                     String champs[] = new String[] { editionNom.getText().toString(), editionDescription.
00126                     getText().toString(), editionLieu.getText().toString(), editionSuperficie.getText().toString() };
00127                     List<String> parametres = Arrays.asList(champs);
00128                     espaceDeTravail.modifierInformations(parametres);
00129
00130                     finish();
00131                 }
00132             });
00133     }
00138     @Override
00139     public void finish()
00140     {
00141         Log.d(TAG, "finish()");
00142
00143         Intent intent = new Intent();
00144         //intent.putExtra("unEspaceDeTravail", espaceDeTravail);
00145         setResult(RESULT_OK, intent);
00146         super.finish();
00147     }
00148
00153     private Handler handler = new Handler()
00154     {
00155         @Override
00156         public void handleMessage(Message msg)
00157         {
00158             super.handleMessage(msg);
00159             Bundle b = msg.getData();
00160
00161             switch(msg.what)
00162             {
00163                 case Communication.TYPE_RECEPTION:
00164                     String trame = b.getString("donnees");

```

```

00165         Log.d(TAG, "handleMessage() Réception [" + b.getString("adresseIP") + ":" + b.getInt("
port") + "] -> " + trame);
00166
00167         String[] champs = trame.split(";");
00168         int typeTrame = Communication.
recupererTypeTrame(champs);
00169         Log.d(TAG, "handleMessage() : typeTrame : " + typeTrame);
00170
00171         break;
00172     default:
00173         Log.d(TAG, "handleMessage() : code inconnu ! ");
00174     }
00175 }
00176 };
00177 }

```

9.13 Référence du fichier Planification.md

9.14 Planification.md

```

00001 # Planification
00002
00003 | Fonctionnalité
00004
00005 |-----|
00006 | Visualiser la liste des espaces de travail
00007 | Haute | 1 |
00008 | Visualiser un espace de travail (nom, lieu, description, superficie, disponibilité, durée
d'occupation, température et indice de confort) | Haute | 1 |
00009 | Communiquer avec les portiers
00010 | Haute | 2 |
00011 | Réserver un espace de travail avec durée d'occupation
00012 | Moyenne | 2 |
00013 | Augmenter la durée d'occupation
00014 | Moyenne | 2 |
00015 | Libérer un espace de travail
00016 | Haute | 2 |
00017 | Rechercher un espace de travail par nom, disponibilité et/ou indice de confort
00018 | Moyenne | 3 |
00019 | Editer un espace de travail
00020 | Haute | 3 |
00021 | Visualiser les favoris
00022 | Basse | 3 |
00023 | Ajouter/retirer un espace de travail aux favoris
00024 | Basse | 3 |
00025
00026 ## Itération 1
00027
00028 * Visualiser la liste des espaces de travail
00029 * Visualiser un espace de travail (nom, lieu, description, superficie, disponibilité, température et
indice de confort)
00030
00031 ## Itération 2
00032
00033 * Communiquer avec les portiers
00034 * Réserver un espace de travail avec durée d'occupation
00035 * Augmenter la durée d'occupation
00036 * Libérer un espace de travail
00037
00038 ## Itération 3
00039
00040 * Rechercher un espace de travail par nom, disponibilité et/ou indice de confort
00041 * Editer un espace de travail
00042 * Visualiser les favoris
00043 * Ajouter/retirer un espace de travail aux favoris

```

9.15 Référence du fichier Protocole.md

9.16 Protocole.md

```

00001 # Protocole Meeting 2021
00002
00003 ## Description générale
00004
00005 Protocole orienté ASCII
00006
00007 Délimiteurs :

```



```

00008
00009 - début : '$'
00010 - champs : ';'
00011 - fin : '\r\n'
00012
00013 Adresse de multicast des portiers : '239.0.0.x'
00014
00015 Format des trames de requêtes Application -> Portier(s) en multicast : '$GET;idRequete\r\n'
00016
00017 Format des trames d'actualisations Application -> Portier(s) en unicast : '$SET;idRequete\r\n'
00018
00019 | idRequete | Signification | Requetes | Actualisations |
00020 |-----|-----|-----|-----|
00021 | '1' | informations | X | X |
00022 | '2' | état | X | |
00023 | '3' | disponibilité | X | X |
00024
00025 Les différents champs d'une trame de réponse :
00026
00027 | champ | description |
00028 |-----|-----|
00029 | nomSalle | string (le nom de la salle) |
00030 | description | string (description de la salle) |
00031 | emplacement | string (l'emplacement de la salle) |
00032 | disponibilité | '0' = occupé et '1' = libre |
00033 | niveauDeConfort | de -3 à 3 (voir cahier des charges) |
00034 | température | en degré |
00035 | surface | en m² |
00036
00037 ### Demande les informations des portiers
00038
00039 Application -> Portier(s) en multicast : '$GET;1\r\n'
00040
00041 Portier -> Application :
00042 '$nomSalle;description;emplacement;surface;disponibilité;niveauDeConfort;température\r\n'
00043 - Nombre de champs : 7
00044
00045 ### Demande l'état des portiers
00046
00047 Application -> Portier(s) en multicast : '$GET;2\r\n'
00048
00049 Portier -> Application : '$nomSalle;disponibilité;niveauDeConfort;température\r\n'
00050
00051 - Nombre de champs : 4
00052
00053 ### Demande la disponibilité des portiers
00054
00055 Application -> Portier(s) en multicast : '$GET;3\r\n'
00056
00057 Portier -> Application : '$nomSalle;disponibilité\r\n'
00058
00059 - Nombre de champs : 2
00060
00061 ### Actualiser les informations d'un portier
00062
00063 Application -> Portier en unicast : '$SET;1;nomSalle;description;emplacement;surface\r\n'
00064
00065 Portier -> Application : '$nomSalle;ok\r\n'
00066
00067 - Nombre de champs : 2
00068
00069 ### Actualiser la disponibilité d'un portier
00070
00071 - Pour réserver :
00072
00073 Application -> Portier en unicast : '$SET;3;0\r\n'
00074
00075 Retour :
00076
00077 Portier -> Application : '$nomSalle;code;OK\r\n'
00078
00079 Nombre de champs : 4
00080
00081 Portier -> Application : '$nomSalle;;ERREUR\r\n'
00082
00083 Nombre de champs : 3
00084
00085 - Pour libérer :
00086
00087 Application -> Portier en unicast : '$SET;3;1;code\r\n'
00088
00089 Portier -> Application : '$nomSalle;;OK\r\n'
00090
00091 Nombre de champs : 3
00092
00093 Portier -> Application : '$nomSalle;;ERREUR\r\n'
00094
00095 Nombre de champs : 3

```

9.17 Référence du fichier README.md

9.18 README.md

```

00001 \mainpage Application Android
00002
00003 \tableofcontents
00004
00005 \section section_tdm Table des matières
00006 - \ref page_README
00007 - \ref page_about
00008 - \ref page_licence
00009
00010 \image html screenshot-meeting.png
00011
00012 \section section_infos Informations
00013
00014 \author KELLER--LAVALLEE Joachim <<joachim.kellerlavallee@gmail.com>>
00015 \date 2021
00016 \version 1.1
00017 \see https://svn.riouxsvn.com/meeting-2021/
00018
00019
00020 \page page_README README
00021
00022 [TOC]
00023
00024 # Meeting {#projet}
00025
00026 ## Présentation {#presentation}
00027
00028 Meeting est une application mobile qui communique avec un portier connecté via une liaison wifi et qui
    permet de :
00029
00030 * Visualiser la liste des espaces de travail
00031 * Visualiser les données de l'espace de travail
00032 * Réserver et libérer un espace de travail
00033 * Editer un espace de travail
00034 * Rechercher un espace de travail par mot-clé, disponibilité et niveau de confort
00035 * Gérer les favoris
00036
00037 \image html screenshot-meeting-1.png
00038
00039 \image html screenshot-meeting-2.png
00040
00041 ## Informations {#informations}
00042
00043 \author KELLER--LAVALLEE Joachim <<joachim.kellerlavallee@gmail.com>>
00044 \date 2021
00045 \version 1.1
00046 \see https://svn.riouxsvn.com/meeting-2021/
00047
00048
00049 \page page_about A propos
00050
00051 \author KELLER--LAVALLEE Joachim <<joachim.kellerlavallee@gmail.com>>
00052
00053
00054 \page page_licence Licence GPL
00055
00056 This program is free software; you can redistribute it and/or modify
00057 it under the terms of the GNU General Public License as published by
00058 the Free Software Foundation; either version 2 of the License, or
00059 (at your option) any later version.
00060
00061 This program is distributed in the hope that it will be useful,
00062 but WITHOUT ANY WARRANTY; without even the implied warranty of
00063 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00064 GNU General Public License for more details.
00065
00066 You should have received a copy of the GNU General Public License
00067 along with this program; if not, write to the Free Software
00068 Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

```

Index

adaptateur
 com : :lasalle : :meeting : :IHMMeting, [72](#)

adresseIP
 com : :lasalle : :meeting : :Communication, [26](#)
 com : :lasalle : :meeting : :EspaceDeTravail, [44](#)

adresseMulticast
 com : :lasalle : :meeting : :Communication, [26](#)

AffichageEspaceDeTravail.java, [83](#)

afficher
 com : :lasalle : :meeting : :AffichageEspaceDeTravail, [8](#)

afficherAdresseIP
 com : :lasalle : :meeting : :AffichageEspaceDeTravail, [9](#)

afficherBoiteLiberation
 com : :lasalle : :meeting : :AffichageEspaceDeTravail, [9](#)

afficherBoutons
 com : :lasalle : :meeting : :AffichageEspaceDeTravail, [10](#)
 com : :lasalle : :meeting : :ModificationEspaceDeTravail, [78](#)

afficherDescription
 com : :lasalle : :meeting : :AffichageEspaceDeTravail, [11](#)

afficherDisponibilite
 com : :lasalle : :meeting : :AffichageEspaceDeTravail, [12](#)

afficherEditionDescription
 com : :lasalle : :meeting : :ModificationEspaceDeTravail, [78](#)

afficherEditionLieu
 com : :lasalle : :meeting : :ModificationEspaceDeTravail, [79](#)

afficherEditionNom
 com : :lasalle : :meeting : :ModificationEspaceDeTravail, [79](#)

afficherEditionSuperficie
 com : :lasalle : :meeting : :ModificationEspaceDeTravail, [79](#)

afficherFavori
 com : :lasalle : :meeting : :AffichageEspaceDeTravail, [12](#)

afficherIndiceDeConfort
 com : :lasalle : :meeting : :AffichageEspaceDeTravail, [12](#)

afficherLieu
 com : :lasalle : :meeting : :AffichageEspaceDeTravail, [13](#)

afficherListeEspacesDeTravail
 com : :lasalle : :meeting : :IHMMeting, [54](#)

afficherNom
 com : :lasalle : :meeting : :AffichageEspaceDeTravail, [13](#)

afficherSuperficie
 com : :lasalle : :meeting : :AffichageEspaceDeTravail, [14](#)

afficherTemperature
 com : :lasalle : :meeting : :AffichageEspaceDeTravail, [14](#)

ajouterEspaceDeTravail
 com : :lasalle : :meeting : :IHMMeting, [55](#)

arreter
 com : :lasalle : :meeting : :Communication, [20](#)

boiteDeDialogueAPropos
 com : :lasalle : :meeting : :IHMMeting, [72](#)

boiteDeDialogueFiltrerParDisponibilite
 com : :lasalle : :meeting : :IHMMeting, [72](#)

boiteDeDialogueFiltrerParNiveauDeConfort
 com : :lasalle : :meeting : :IHMMeting, [73](#)

boiteDeDialogueRechercher
 com : :lasalle : :meeting : :IHMMeting, [73](#)

CHAMP_CODE

 com : :lasalle : :meeting : :Communication, [26](#)

CHAMP_DESCRIPTION
 com : :lasalle : :meeting : :Communication, [26](#)

CHAMP_DISPONIBILITE
 com : :lasalle : :meeting : :Communication, [27](#)

CHAMP_INDICE_DE_CONFORT
 com : :lasalle : :meeting : :Communication, [27](#)

CHAMP_LIEU
 com : :lasalle : :meeting : :Communication, [27](#)

CHAMP_NOM
 com : :lasalle : :meeting : :Communication, [27](#)

CHAMP_SUPERFICIE
 com : :lasalle : :meeting : :Communication, [27](#)

CHAMP_TEMPERATURE
 com : :lasalle : :meeting : :Communication, [27](#)

choixFiltre
 com : :lasalle : :meeting : :IHMMeting, [73](#)

code
 com : :lasalle : :meeting : :EspaceDeTravail, [44](#)

com, [5](#)

com.lasalle, [5](#)

com.lasalle.meeting, [6](#)

com.lasalle.meeting.AffichageEspaceDeTravail, [6](#)

com.lasalle.meeting.Communication, [17](#)

com.lasalle.meeting.EspaceDeTravail, [32](#)

com.lasalle.meeting.EspaceDeTravailAdaptateur, [48](#)

com.lasalle.meeting.EspaceDeTravailAdaptateur.ViewHolder, [82](#)

com.lasalle.meeting.IHMMeting, [51](#)

com.lasalle.meeting.ModificationEspaceDeTravail, [77](#)

com : :lasalle : :meeting : :AffichageEspaceDeTravail
 afficher, [8](#)
 afficherAdresseIP, [9](#)
 afficherBoiteLiberation, [9](#)
 afficherBoutons, [10](#)
 afficherDescription, [11](#)
 afficherDisponibilite, [12](#)
 afficherFavori, [12](#)
 afficherIndiceDeConfort, [12](#)
 afficherLieu, [13](#)
 afficherNom, [13](#)
 afficherSuperficie, [14](#)
 afficherTemperature, [14](#)
 espaceDeTravail, [16](#)
 finish, [14](#)
 handler, [16](#)
 onActivityResult, [15](#)
 onCreate, [15](#)
 TAG, [16](#)

com : :lasalle : :meeting : :Communication
 adresseIP, [26](#)
 adresseMulticast, [26](#)
 arreter, [20](#)
 CHAMP_CODE, [26](#)
 CHAMP_DESCRIPTION, [26](#)
 CHAMP_DISPONIBILITE, [27](#)
 CHAMP_INDICE_DE_CONFORT, [27](#)
 CHAMP_LIEU, [27](#)
 CHAMP_NOM, [27](#)

- CHAMP_SUPERFICIE, 27
- CHAMP_TEMPERATURE, 27
- Communication, 19
- DELIMITEUR_CHAMP, 28
- DELIMITEUR_EN_TETE, 28
- DELIMITEUR_FIN, 28
- DEMANDE_DISPONIBILITE, 28
- DEMANDE_INFORMATIONS, 28
- envoyer, 20
- envoyerMessage, 21
- fabriquerTrameDemande, 21
- fabriquerTrameModification, 22
- handler, 28
- MODIFICATION_DISPONIBILITE, 29
- MODIFICATION_INFORMATIONS, 29
- mutex, 29
- NB_CHAMPS_DEMANDE_DISPONIBILITE, 29
- NB_CHAMPS_DEMANDE_INFORMATIONS, 29
- NB_CHAMPS_DISPONIBILITE_CODE, 30
- NB_CHAMPS_DISPONIBILITE, 30
- NB_CHAMPS_INFORMATIONS, 30
- NB_CHAMPS_MODIFICATION_DISPONIBILITE, 30
- NB_CHAMPS_RETOUTR Modification_DISPONIBILITE, 30
- PORT, 30
- queueEmission, 31
- recevoir, 23
- recupererTypeTrame, 24
- run, 25
- setHandler, 25
- socket, 31
- TAG, 31
- TRAME_INCONNUE, 31
- TYPE_RECEPTION, 31
- verifierTrame, 25
- com : :lasalle : :meeting : :EspaceDeTravail
 - adressesIP, 44
 - code, 44
 - communication, 45
 - demandeInformations, 34
 - description, 45
 - EspaceDeTravail, 34
 - estFavori, 45
 - estReserve, 45
 - extraireCode, 35
 - extraireInformations, 35
 - fromJSON, 36
 - getAdressesIP, 37
 - getCode, 37
 - getDescription, 37
 - getEstFavori, 38
 - getEstReserve, 38
 - getIndiceDeConfort, 38
 - getLieu, 39
 - getNom, 39
 - getSuperficie, 39
 - getTemperature, 40
 - INDICE_CHAUD, 45
 - INDICE_FRAIS, 46
 - INDICE_FROID, 46
 - INDICE_LEGEREMENT_FRAIS, 46
 - INDICE_LEGEREMENT_TIEDE, 46
 - INDICE_NEUTRE, 46
 - INDICE_TIEDE, 47
 - indiceDeConfort, 47
 - initialiserCommunication, 40
 - liberer, 41
 - lieu, 47
 - modifierInformations, 41
 - nom, 47
 - reserver, 42
 - setCode, 42
 - setEstFavori, 43
 - setEstReserve, 43
 - superficie, 47
 - TAG, 48
 - temperature, 48
 - toJSON, 44
- com : :lasalle : :meeting : :IHMMMeeting
 - adaptateur, 72
 - afficherListeEspacesDeTravail, 54
 - ajouterEspaceDeTravail, 55
 - boiteDeDialogueAPropos, 72
 - boiteDeDialogueFiltrerParDisponibilite, 72
 - boiteDeDialogueFiltrerParNiveauDeConfort, 73
 - boiteDeDialogueRechercher, 73
 - choixFiltre, 73
 - communication, 73
 - containsIgnoreCase, 55
 - demarrerReseau, 56
 - espacesDeTravail, 73
 - espacesDeTravailFiltres, 74
 - estReserve, 74
 - filtreParDisponibilite, 56
 - filtreParFavoris, 57
 - filtreParNiveauDeConfort, 57
 - filtreTous, 58
 - handler, 74
 - indiceDeConfort, 74
 - initialiserBoiteDeDialogueFiltrerParDisponibilite, 58
 - initialiserBoiteDeDialogueFiltrerParNiveauDeConfort, 59
 - initialiserBoiteDeDialogueRechercher, 60
 - initialiserBoitesDeDialogue, 61
 - initialiserEspacesDeTravail, 61
 - initialiserListeEspacesDeTravail, 62
 - initialiserPreferences, 62
 - listeEspacesDeTravail, 74
 - modifierEspaceDeTravail, 62
 - motCle, 75
 - onActivityResult, 63
 - onCreate, 63
 - onCreateOptionsMenu, 64
 - onDestroy, 64
 - onOptionsItemSelected, 64
 - onPause, 65
 - onRadioButtonClicked, 65
 - onResume, 66
 - onStart, 67

- onStop, [67](#)
- PREFERENCES_CODE, [75](#)
- PREFERENCES_EST_FAVORI, [75](#)
- PREFERENCES, [75](#)
- preferences, [75](#)
- rafraichirAffichageListeEspaces, [67](#)
- rafraichirListeEspacesFiltres, [68](#)
- rechercher, [68](#)
- recupererDonneesEspaceDeTravail, [69](#)
- sauvegarderDonneesEspaceDeTravail, [69](#)
- supprimerEspaceDeTravail, [70](#)
- swipeRefreshLayout, [76](#)
- TAG, [76](#)
- titreEspacesDeTravail, [76](#)
- trierEspacesDeTravail, [70](#)
- verifierPresenceEspaceDeTravail, [71](#)
- wm, [76](#)
- com : :lasalle : :meeting : :ModificationEspaceDeTravail
 - afficherBoutons, [78](#)
 - afficherEditionDescription, [78](#)
 - afficherEditionLieu, [79](#)
 - afficherEditionNom, [79](#)
 - afficherEditionSuperficie, [79](#)
 - espaceDeTravail, [81](#)
 - finish, [80](#)
 - handler, [81](#)
 - onCreate, [80](#)
 - TAG, [81](#)
- Communication
 - com : :lasalle : :meeting : :Communication, [19](#)
- communication
 - com : :lasalle : :meeting : :EspaceDeTravail, [45](#)
 - com : :lasalle : :meeting : :IHMMMeeting, [73](#)
- Communication.java, [88](#)
- containsIgnoreCase
 - com : :lasalle : :meeting : :IHMMMeeting, [55](#)
- DELIMITEUR_CHAMP
 - com : :lasalle : :meeting : :Communication, [28](#)
- DELIMITEUR_EN_TETE
 - com : :lasalle : :meeting : :Communication, [28](#)
- DELIMITEUR_FIN
 - com : :lasalle : :meeting : :Communication, [28](#)
- DEMANDE_DISPONIBILITE
 - com : :lasalle : :meeting : :Communication, [28](#)
- DEMANDE_INFORMATIONS
 - com : :lasalle : :meeting : :Communication, [28](#)
- demanderInformations
 - com : :lasalle : :meeting : :EspaceDeTravail, [34](#)
- demarrerReseau
 - com : :lasalle : :meeting : :IHMMMeeting, [56](#)
- description
 - com : :lasalle : :meeting : :EspaceDeTravail, [45](#)
- envoyer
 - com : :lasalle : :meeting : :Communication, [20](#)
- envoyerMessage
 - com : :lasalle : :meeting : :Communication, [21](#)
- EspaceDeTravail
 - com : :lasalle : :meeting : :EspaceDeTravail, [34](#)
- espaceDeTravail
 - com : :lasalle : :meeting : :AffichageEspaceDeTravail, [16](#)
 - com : :lasalle : :meeting : :ModificationEspaceDeTravail, [81](#)
- EspaceDeTravail.java, [92, 93](#)
- EspaceDeTravailAdaptateur
 - com : :lasalle : :meeting : :EspaceDeTravailAdaptateur, [49](#)
- EspaceDeTravailAdaptateur.java, [96, 97](#)
- espacesDeTravail
 - com : :lasalle : :meeting : :IHMMMeeting, [73](#)
- espacesDeTravailFiltres
 - com : :lasalle : :meeting : :IHMMMeeting, [74](#)
- estFavori
 - com : :lasalle : :meeting : :EspaceDeTravail, [45](#)
- estReserve
 - com : :lasalle : :meeting : :EspaceDeTravail, [45](#)
 - com : :lasalle : :meeting : :IHMMMeeting, [74](#)
- extraireCode
 - com : :lasalle : :meeting : :EspaceDeTravail, [35](#)
- extraireInformations
 - com : :lasalle : :meeting : :EspaceDeTravail, [35](#)
- fabriquerTrameDemande
 - com : :lasalle : :meeting : :Communication, [21](#)
- fabriquerTrameModification
 - com : :lasalle : :meeting : :Communication, [22](#)
- filtrerParDisponibilite
 - com : :lasalle : :meeting : :IHMMMeeting, [56](#)
- filtrerParFavoris
 - com : :lasalle : :meeting : :IHMMMeeting, [57](#)
- filtrerParNiveauDeConfort
 - com : :lasalle : :meeting : :IHMMMeeting, [57](#)
- filtrerTous
 - com : :lasalle : :meeting : :IHMMMeeting, [58](#)
- finish
 - com : :lasalle : :meeting : :AffichageEspaceDeTravail, [14](#)
 - com : :lasalle : :meeting : :ModificationEspaceDeTravail, [80](#)
- fromJSON
 - com : :lasalle : :meeting : :EspaceDeTravail, [36](#)
- getAdresselP
 - com : :lasalle : :meeting : :EspaceDeTravail, [37](#)
- getCode
 - com : :lasalle : :meeting : :EspaceDeTravail, [37](#)
- getDescription
 - com : :lasalle : :meeting : :EspaceDeTravail, [37](#)
- getEstFavori
 - com : :lasalle : :meeting : :EspaceDeTravail, [38](#)
- getEstReserve
 - com : :lasalle : :meeting : :EspaceDeTravail, [38](#)
- getIndiceDeConfort
 - com : :lasalle : :meeting : :EspaceDeTravail, [38](#)
- getLieu
 - com : :lasalle : :meeting : :EspaceDeTravail, [39](#)
- getNom
 - com : :lasalle : :meeting : :EspaceDeTravail, [39](#)
- getSuperficie
 - com : :lasalle : :meeting : :EspaceDeTravail, [39](#)
- getTemperature
 - com : :lasalle : :meeting : :EspaceDeTravail, [40](#)
- getView
 - com : :lasalle : :meeting : :EspaceDeTravailAdaptateur, [49](#)
- handler

- com : :lasalle : :meeting : :AffichageEspaceDeTravail, 16
- com : :lasalle : :meeting : :Communication, 28
- com : :lasalle : :meeting : :IHMMeting, 74
- com : :lasalle : :meeting : :ModificationEspaceDeTravail, 81
- IHMMeeting.java, 98, 99
- INDICE_CHAUD
 - com : :lasalle : :meeting : :EspaceDeTravail, 45
- INDICE_FRAIS
 - com : :lasalle : :meeting : :EspaceDeTravail, 46
- INDICE_FROID
 - com : :lasalle : :meeting : :EspaceDeTravail, 46
- INDICE_LEGEREMENT_FRAIS
 - com : :lasalle : :meeting : :EspaceDeTravail, 46
- INDICE_LEGEREMENT_TIEDE
 - com : :lasalle : :meeting : :EspaceDeTravail, 46
- INDICE_NEUTRE
 - com : :lasalle : :meeting : :EspaceDeTravail, 46
- INDICE_TIEDE
 - com : :lasalle : :meeting : :EspaceDeTravail, 47
- indiceDeConfort
 - com : :lasalle : :meeting : :EspaceDeTravail, 47
 - com : :lasalle : :meeting : :IHMMeting, 74
- initialiserBoiteDeDialogueFiltrerParDisponibilite
 - com : :lasalle : :meeting : :IHMMeting, 58
- initialiserBoiteDeDialogueFiltrerParNiveauDeConfort
 - com : :lasalle : :meeting : :IHMMeting, 59
- initialiserBoiteDeDialogueRechercher
 - com : :lasalle : :meeting : :IHMMeting, 60
- initialiserBoitesDeDialogue
 - com : :lasalle : :meeting : :IHMMeting, 61
- initialiserCommunication
 - com : :lasalle : :meeting : :EspaceDeTravail, 40
- initialiserEspacesDeTravail
 - com : :lasalle : :meeting : :IHMMeting, 61
- initialiserListeEspacesDeTravail
 - com : :lasalle : :meeting : :IHMMeting, 62
- initialiserPreferences
 - com : :lasalle : :meeting : :IHMMeting, 62
- liberer
 - com : :lasalle : :meeting : :EspaceDeTravail, 41
- lieu
 - com : :lasalle : :meeting : :EspaceDeTravail, 47
- listeEspacesDeTravail
 - com : :lasalle : :meeting : :IHMMeting, 74
- MODIFICATION_DISPONIBILITE
 - com : :lasalle : :meeting : :Communication, 29
- MODIFICATION_INFORMATIONS
 - com : :lasalle : :meeting : :Communication, 29
- ModificationEspaceDeTravail.java, 108
- modifierEspaceDeTravail
 - com : :lasalle : :meeting : :IHMMeting, 62
- modifierInformations
 - com : :lasalle : :meeting : :EspaceDeTravail, 41
- motCle
 - com : :lasalle : :meeting : :IHMMeting, 75
- mutex
 - com : :lasalle : :meeting : :Communication, 29
- NB_CHAMPS_DEMANDE_DISPONIBILITE
 - com : :lasalle : :meeting : :Communication, 29
- NB_CHAMPS_DEMANDE_INFORMATIONS
 - com : :lasalle : :meeting : :Communication, 29
- NB_CHAMPS_DISPONIBILITE_CODE
 - com : :lasalle : :meeting : :Communication, 30
- NB_CHAMPS_DISPONIBILITE
 - com : :lasalle : :meeting : :Communication, 30
- NB_CHAMPS_INFORMATIONS
 - com : :lasalle : :meeting : :Communication, 30
- NB_CHAMPS_MODIFICATION_DISPONIBILITE
 - com : :lasalle : :meeting : :Communication, 30
- NB_CHAMPS_RETOUR_MODIFICATION_DISPONIBILITE
 - com : :lasalle : :meeting : :Communication, 30
- nom
 - com : :lasalle : :meeting : :EspaceDeTravail, 47
- onActivityResult
 - com : :lasalle : :meeting : :AffichageEspaceDeTravail, 15
 - com : :lasalle : :meeting : :IHMMeting, 63
- onCreate
 - com : :lasalle : :meeting : :AffichageEspaceDeTravail, 15
 - com : :lasalle : :meeting : :IHMMeting, 63
 - com : :lasalle : :meeting : :ModificationEspaceDeTravail, 80
- onCreateOptionsMenu
 - com : :lasalle : :meeting : :IHMMeting, 64
- onDestroy
 - com : :lasalle : :meeting : :IHMMeting, 64
- onOptionsItemSelected
 - com : :lasalle : :meeting : :IHMMeting, 64
- onPause
 - com : :lasalle : :meeting : :IHMMeting, 65
- onRadioButtonClicked
 - com : :lasalle : :meeting : :IHMMeting, 65
- onResume
 - com : :lasalle : :meeting : :IHMMeting, 66
- onStart
 - com : :lasalle : :meeting : :IHMMeting, 67
- onStop
 - com : :lasalle : :meeting : :IHMMeting, 67
- PORT
 - com : :lasalle : :meeting : :Communication, 30
- PREFERENCES_CODE
 - com : :lasalle : :meeting : :IHMMeting, 75
- PREFERENCES_EST_FAVORI
 - com : :lasalle : :meeting : :IHMMeting, 75
- PREFERENCES
 - com : :lasalle : :meeting : :IHMMeting, 75
- Planification.md, 110
- preferences
 - com : :lasalle : :meeting : :IHMMeting, 75
- Protocole.md, 110
- queueEmission
 - com : :lasalle : :meeting : :Communication, 31
- README.md, 112
- rafraichirAffichageListeEspaces
 - com : :lasalle : :meeting : :IHMMeting, 67
- rafraichirListeEspacesFiltres
 - com : :lasalle : :meeting : :IHMMeting, 68
- recevoir

- com : :lasalle : :meeting : :Communication, [23](#)
- rechercher
 - com : :lasalle : :meeting : :IHMMMeeting, [68](#)
- recupererDonneesEspaceDeTravail
 - com : :lasalle : :meeting : :IHMMMeeting, [69](#)
- recupererTypeTrame
 - com : :lasalle : :meeting : :Communication, [24](#)
- reserver
 - com : :lasalle : :meeting : :EspaceDeTravail, [42](#)
- run
 - com : :lasalle : :meeting : :Communication, [25](#)
- Runnable, [82](#)
- sauvegarderDonneesEspaceDeTravail
 - com : :lasalle : :meeting : :IHMMMeeting, [69](#)
- setCode
 - com : :lasalle : :meeting : :EspaceDeTravail, [42](#)
- setEstFavori
 - com : :lasalle : :meeting : :EspaceDeTravail, [43](#)
- setEstReserve
 - com : :lasalle : :meeting : :EspaceDeTravail, [43](#)
- setHandler
 - com : :lasalle : :meeting : :Communication, [25](#)
- socket
 - com : :lasalle : :meeting : :Communication, [31](#)
- superficie
 - com : :lasalle : :meeting : :EspaceDeTravail, [47](#)
- supprimerEspaceDeTravail
 - com : :lasalle : :meeting : :IHMMMeeting, [70](#)
- swipeRefreshLayout
 - com : :lasalle : :meeting : :IHMMMeeting, [76](#)
- TAG
 - com : :lasalle : :meeting : :AffichageEspaceDeTravail, [16](#)
 - com : :lasalle : :meeting : :Communication, [31](#)
 - com : :lasalle : :meeting : :EspaceDeTravail, [48](#)
 - com : :lasalle : :meeting : :EspaceDeTravailAdaptateur, [51](#)
 - com : :lasalle : :meeting : :IHMMMeeting, [76](#)
 - com : :lasalle : :meeting : :ModificationEspaceDeTravail, [81](#)
- TRAME_INCONNUE
 - com : :lasalle : :meeting : :Communication, [31](#)
- TYPE_RECEPTION
 - com : :lasalle : :meeting : :Communication, [31](#)
- temperature
 - com : :lasalle : :meeting : :EspaceDeTravail, [48](#)
- titreEspacesDeTravail
 - com : :lasalle : :meeting : :IHMMMeeting, [76](#)
- toJSON
 - com : :lasalle : :meeting : :EspaceDeTravail, [44](#)
- trierEspacesDeTravail
 - com : :lasalle : :meeting : :IHMMMeeting, [70](#)
- verifierPresenceEspaceDeTravail
 - com : :lasalle : :meeting : :IHMMMeeting, [71](#)
- verifierTrame
 - com : :lasalle : :meeting : :Communication, [25](#)
- wm
 - com : :lasalle : :meeting : :IHMMMeeting, [76](#)