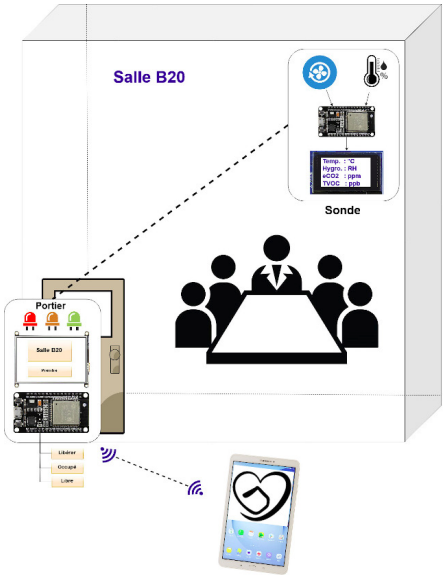
KELLER--LAVALLEE Joachim
BTS SNIR - Session 2021

Dossier technique

Projet Meeting

Sommaire

Présentation du projet



La gestion des espaces de travail et le confort sont des problématiques importantes pour les entreprises.

Le système Meeting est composé d'un portier placé à l'extérieur d'un espace de travail. Il permet d'afficher les informations de l'espace de travail concerné (nom, disponibilité, température, indice de confort) et de le réserver. Le portier communique avec une sonde placée à l'intérieur de l'espace de travail et avec une application mobile.
L'application mobile permet d'afficher tous les espaces de travail d'un bâtiment, de les réserver et de les ajouter aux favoris.

Le système doit permettre de :

- Afficher et signaler la disponibilité de l'espace de travail
- Afficher les informations, la température et l'indice de confort de l'espace de travail
- Réserver et libérer l'espace de travail sur l'écran tactile du portier ou depuis l'application mobile sur un smartphone

Etudiants en charge du projet

- Option EC :
  - Etudiant 1 : GUILLEMIN Kerloye
  - Etudiant 2 : SAINT-JEAN Charlotte
- Option IR :
  - Etudiant 3 : KELLER--LAVALLEE Joachim

Travail à réaliser

Il faut réaliser une application mobile Android codée en Java.
L'application permet à l'utilisateur de :

- Voir la liste des espaces de travail
- Voir les données associées à chaque espace de travail :
    - Nom
    - Lieu
    - Description
    - Superficie
    - Température
    - Indice de confort
    - Disponibilité
- Réserver un espace de travail
- Libérer un espace de travail
- Editer les informations d'un espace de travail

Ressources logicielles :

- Environnement de développement : Android Studio 4.2
- Gestionnaire de version : Subversion 1.13.0
- Générateur de diagrammes UML : BOUML 7.11
- Générateur de documentation : Doxygen 1.8.17
- Planification : Beesbusy
- Système d'exploitation du poste de développement : Ubuntu 20.04
- Système d'exploitation de la tablette de test Samsung : Android 7.0

Planification

La planification du projet est gérée avec le logiciel Beesbusy.

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
- Libérer un espace de travail

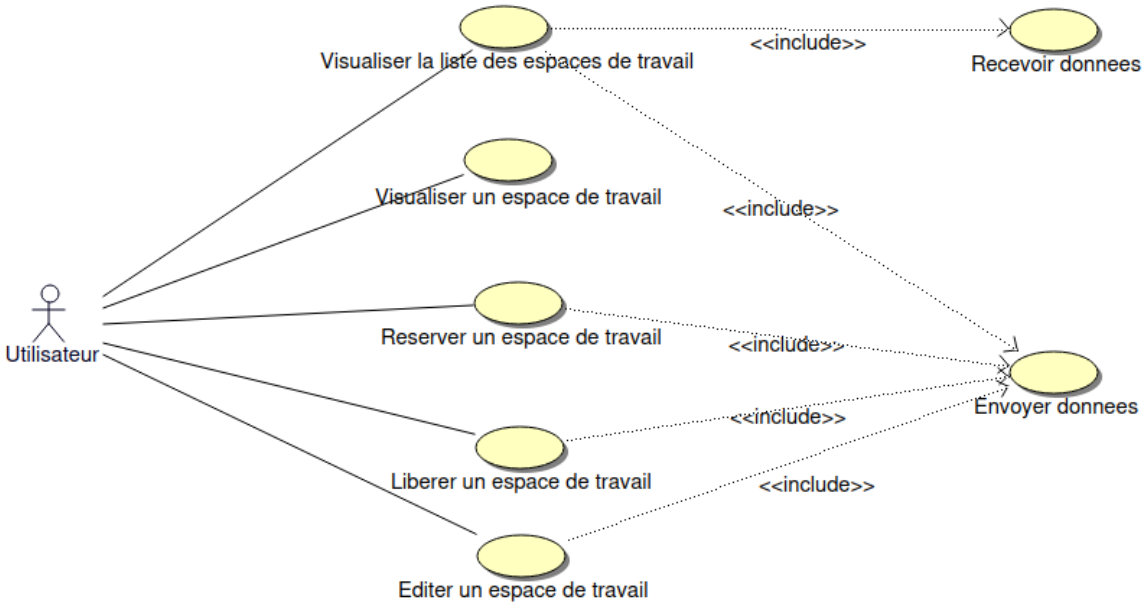
Itération 3 :

- 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

Diagramme des cas d'utilisation

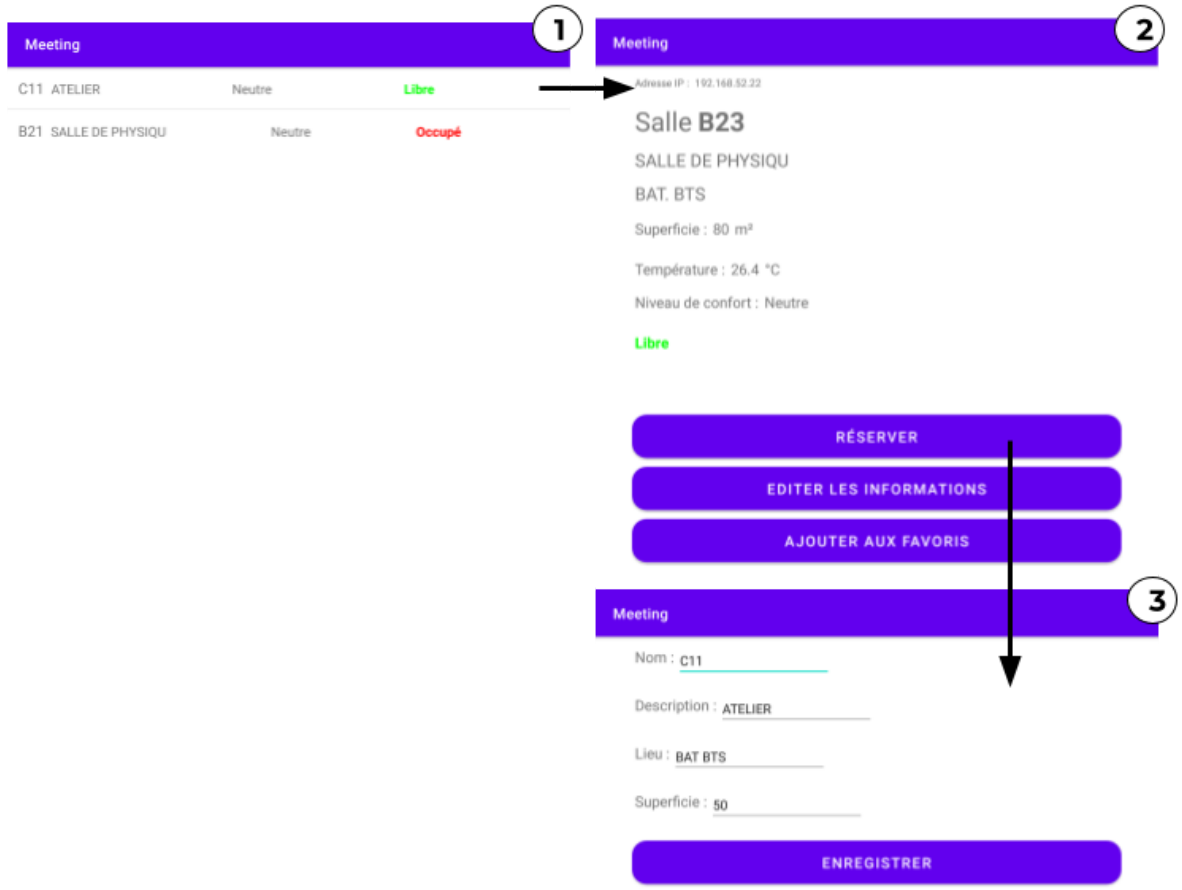
L'utilisateur doit pouvoir :

- Visualiser la liste des espaces de travail
- Visualiser un espace de travail (les données associées à cet espace de travail)
- Réserver un espace de travail
- Libérer un espace de travail
- Editer un espace de travail



Pour afficher la liste des espaces de travail et leurs données, l'application envoie une requête à tous les portiers, puis reçoit les données des portiers.
Pour réserver, libérer, ou éditer un espace de travail, l'application envoie une requête au portier concerné.

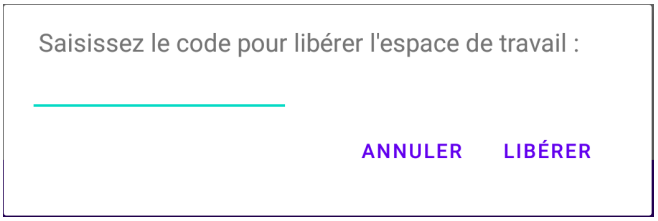
Interface graphique



Sur la **page d'accueil (1)**, la liste des espaces de travail détectés est affichée. Elle peut être mise à jour par Pull To Refresh (une fonctionnalité très utilisée sur Android pour actualiser un contenu).
Cliquer sur un espace de travail ouvre la page d'affichage de cet espace de travail.

Sur la **page d'affichage de l'espace de travail (2)**, les informations, la température, l'indice de confort et la disponibilité sont affichées.
Le bouton "Réserver" ou "Libérer" permet de réserver ou libérer l'espace de travail.
Le bouton "Editer les informations" ouvre la page de modification de l'espace de travail.
Le bouton "Ajouter aux favoris" ou "Retirer des favoris" permet d'ajouter ou de retirer l'espace de travail aux favoris.

Lorsque l'on clique sur le bouton "Libérer", une boîte de dialogue s'affiche. Il faut saisir le code puis cliquer sur "Libérer" pour libérer l'espace de travail.

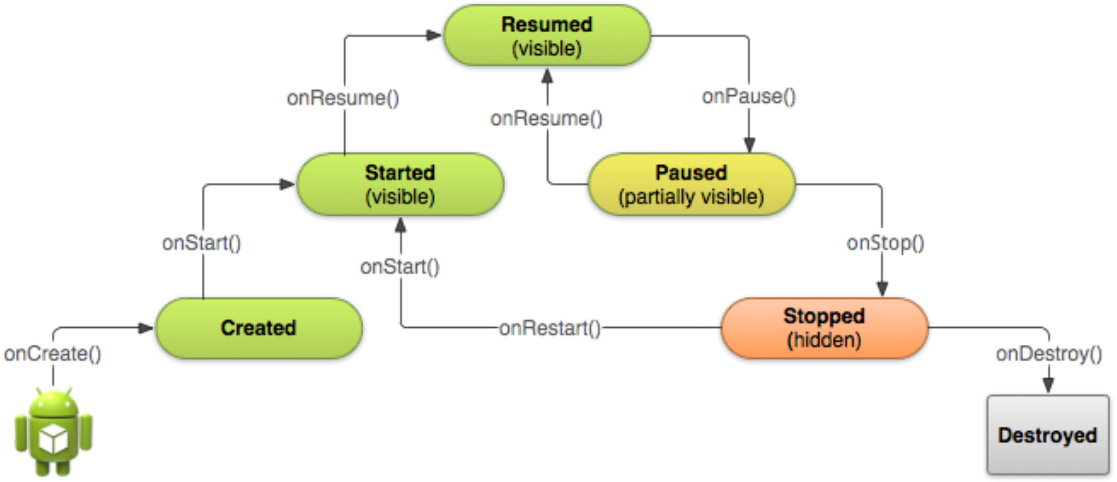


Sur la **page de modification de l'espace de travail (3)**, il est possible de modifier le nom, la description, le lieu et la superficie.
Cliquer sur le bouton "Enregistrer" pour enregistrer les modifications et retourner à la page d'accueil.

Android

Sous Android, une fenêtre (page) d'application est une **activité** (*activity*).
Une activité est composée d'éléments graphiques nommés *widgets* : boutons, vues en liste, champs de saisie, images...

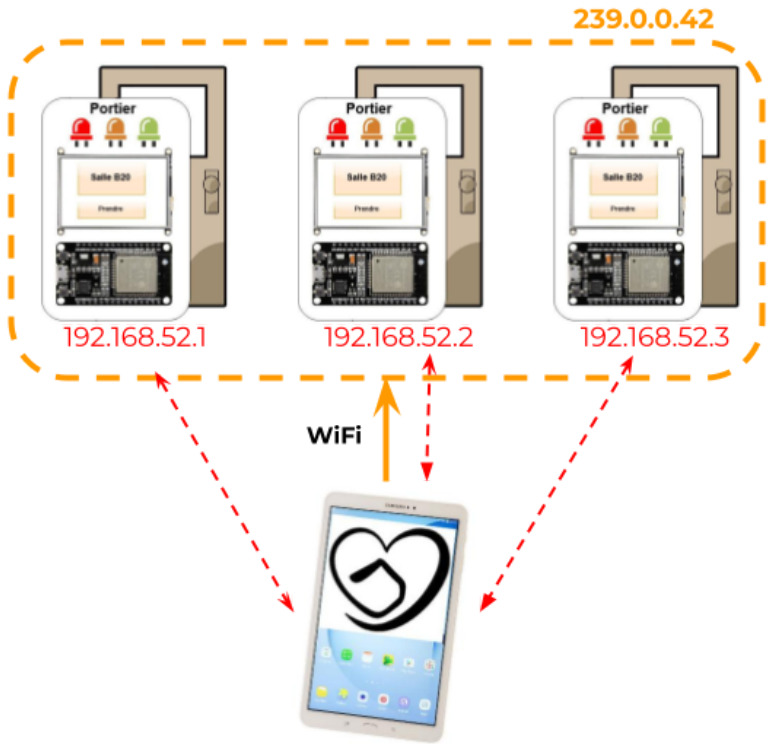
Une activité a un **cycle de vie** géré par le système d'exploitation Android.
Une activité est toujours dans l'un des quatre états suivants : active, en pause, stoppée ou morte. Pour passer d'un état à un autre, les méthodes **`onCreate()`**, **`onPause()`**, **`onResume()`**, **`onRestart()`** et **`onDestroy()`** sont utilisées.



Communication

Entre les portiers et l'application mobile, on utilise une **communication sans fil WiFi**.

Pour communiquer avec les portiers, chaque **portier** a une **adresse IP unicast**. L'**adresse IP multicast** permettant de **communiquer avec l'ensemble des portiers** est : **`239.0.0.42`**.



L'adresse unicast des portiers est utilisée pour la réservation et l'édition d'un espace de travail : l'application envoie une trame en unicast au portier concerné. L'adresse multicast est utilisée pour détecter la présence de portiers : l'application envoie une trame en multicast à tous les portiers, et les portiers répondent chacun en unicast.

Le protocole de la couche Transport est **UDP**. On choisit UDP car il permet le multicast, contrairement à TCP.

Le protocole de la couche Application est propre au projet : voir Protocole Meeting.

Protocole Meeting

Le protocole est orienté ASCII.

Délimiteurs :

- Début : **‘$’**
- Fin : **“\r\n”**
- Champs : **‘;’**

Types de trames :

- Trames de **demande**, envoyées en **multicast** aux portiers :
    - Demande d’**informations** : **`$GET;1\r\n`**
      Le portier répond en unicast :
      `$nom;description;lieu;superficie;disponibilité;indiceDeConfort;température\r\n`

    - Demande de **disponibilité** : **`$GET;3\r\n`**
      Le portier répond en unicast : **`$nom;disponibilité\r\n`**

- Trames de **modification**, envoyées en **unicast** au portier :
    - Modification des **informations** :
      `$SET;1;nom;description;lieu;superficie\r\n`
      Le portier répond en unicast : **`$nom;message\r\n`**

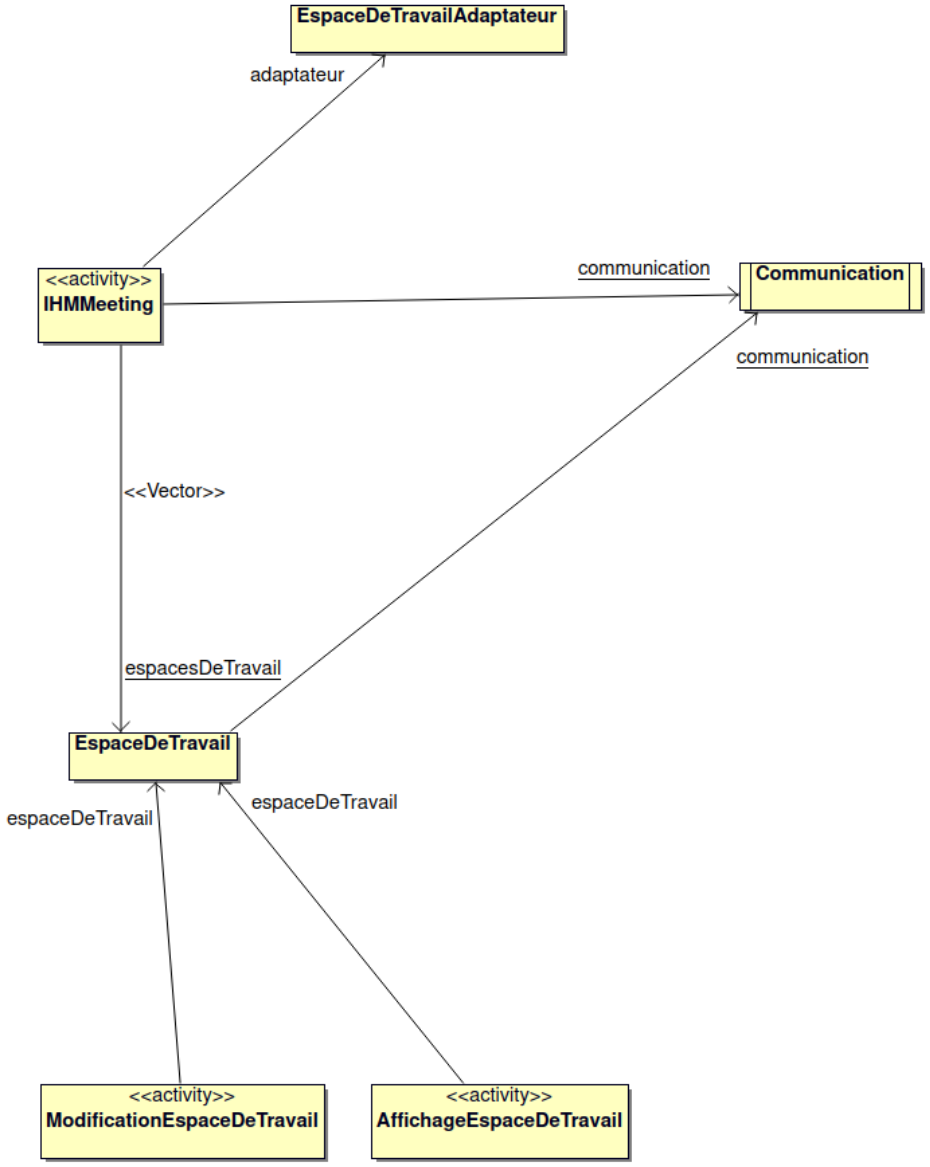
    - Modification de la **disponibilité** :
        - **Réservation : `$SET;3;disponibilité\r\n`**
        - **Libération : `$SET;3;disponibilité;code\r\n`**

      Le portier répond en unicast : **`$nom;code;message\r\n`**

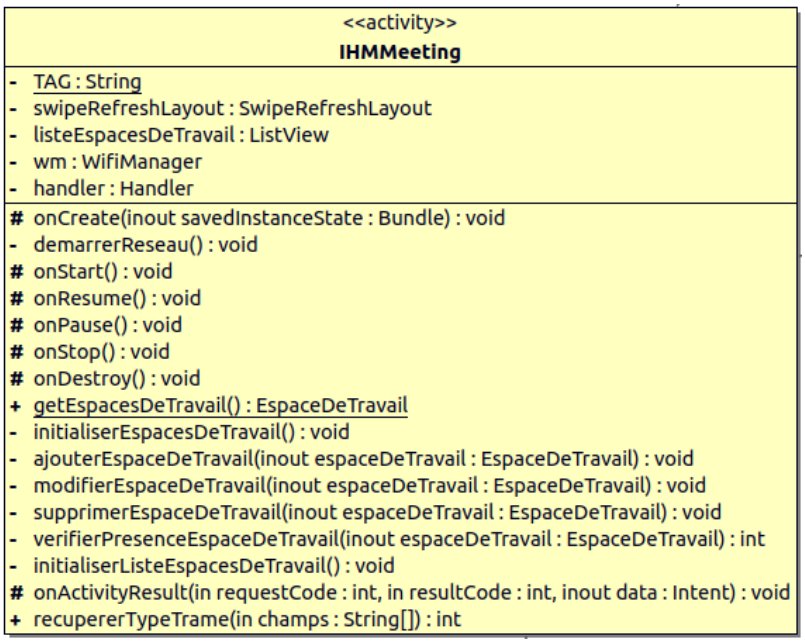
Champs des trames de réponse :

Champ	Description	Valeurs
nom	Nom de l'espace de travail	
description	Description de l'espace de travail	
lieu	Lieu de l'espace de travail	
superficie	Superficie de l'espace de travail en m^2	
disponibilité	Disponibilité de l'espace de travail	'0' : occupé '1' : libre
indiceDeConfort	Indice de confort de l'espace de travail	-3 à 3
température	Température de l'espace de travail en °C	
code	Code pour libérer l'espace de travail	
message	Indique si la modification a été réalisée	"OK" ou "ERREUR"

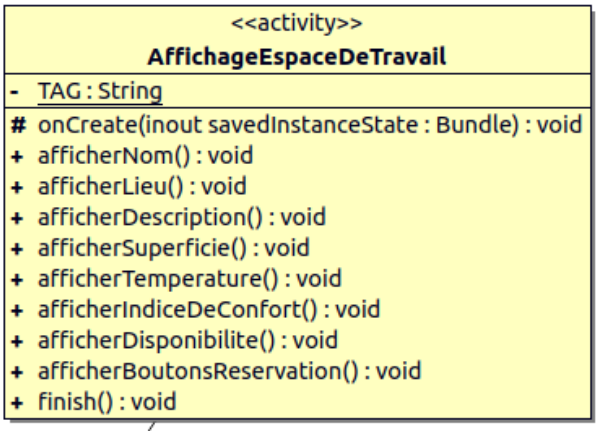
Diagramme de classes



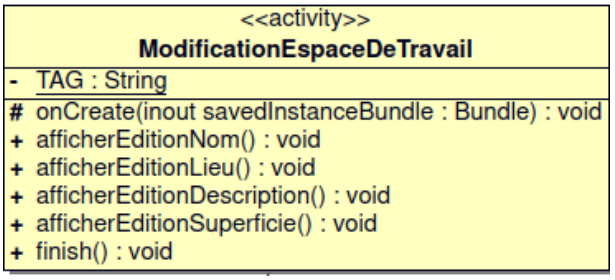
La classe **`IHMMeeting`** gère la page d'accueil.



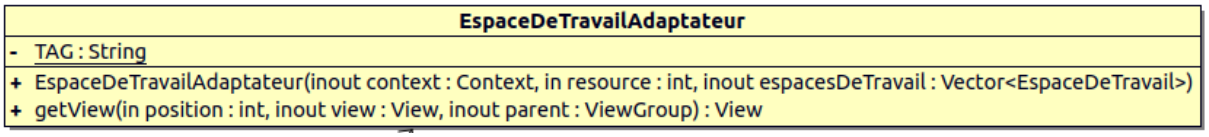
La classe **`AffichageEspaceDeTravail`** gère la page d'affichage de l'espace de travail.



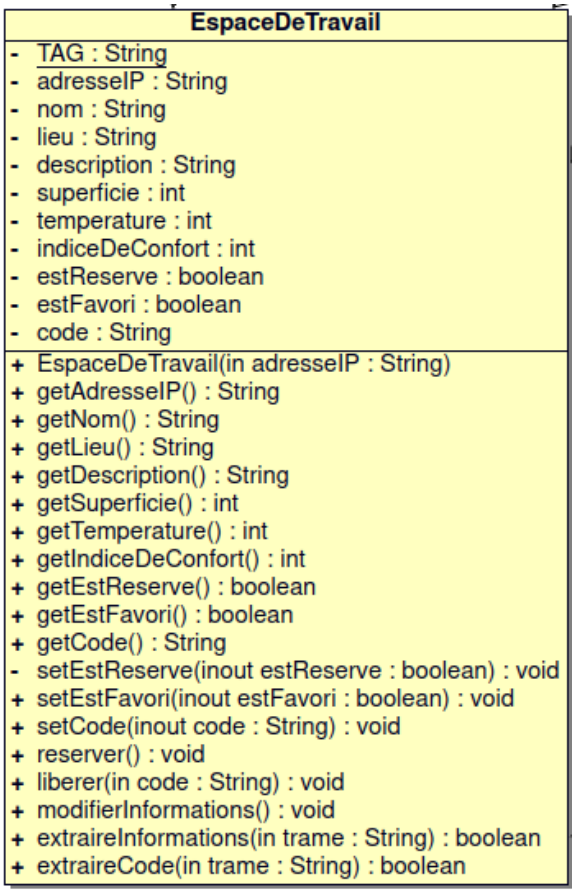
La classe **`ModificationEspaceDeTravail`** gère la page de modification d'un espace de travail.



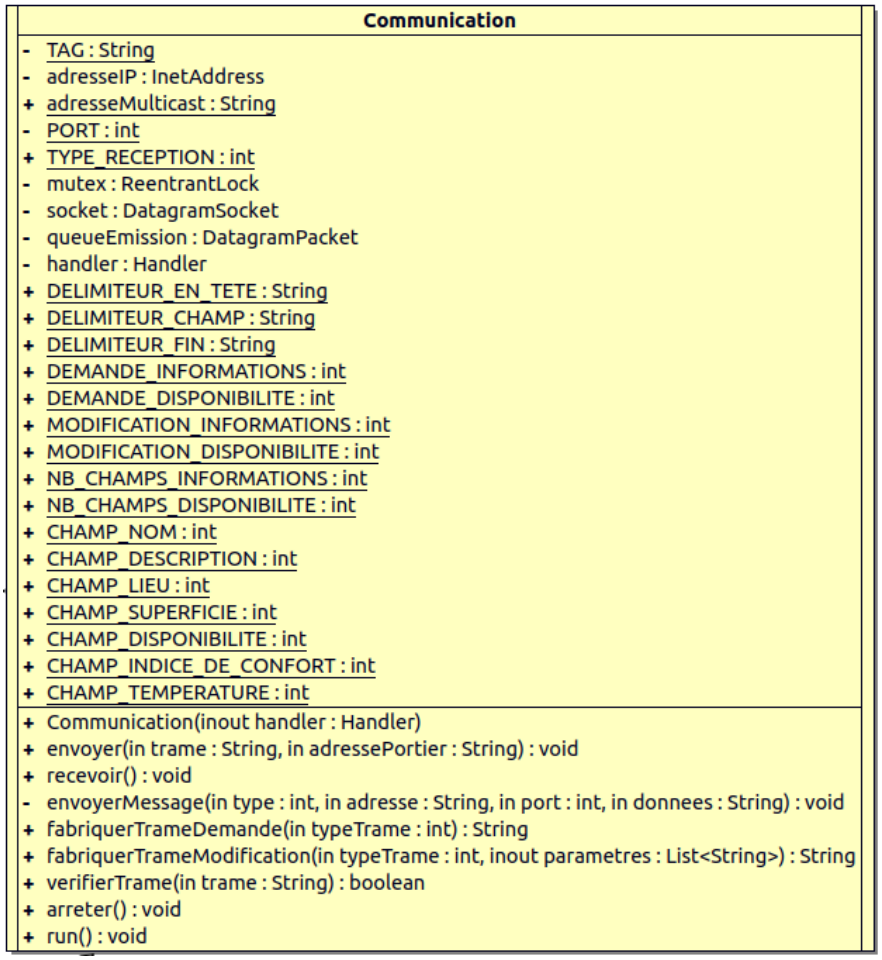
La classe **`EspaceDeTravailAdaptateur`** gère l'affichage des espaces de travail sur la page d'accueil.



La classe **`EspaceDeTravail`** gère l'espace de travail.



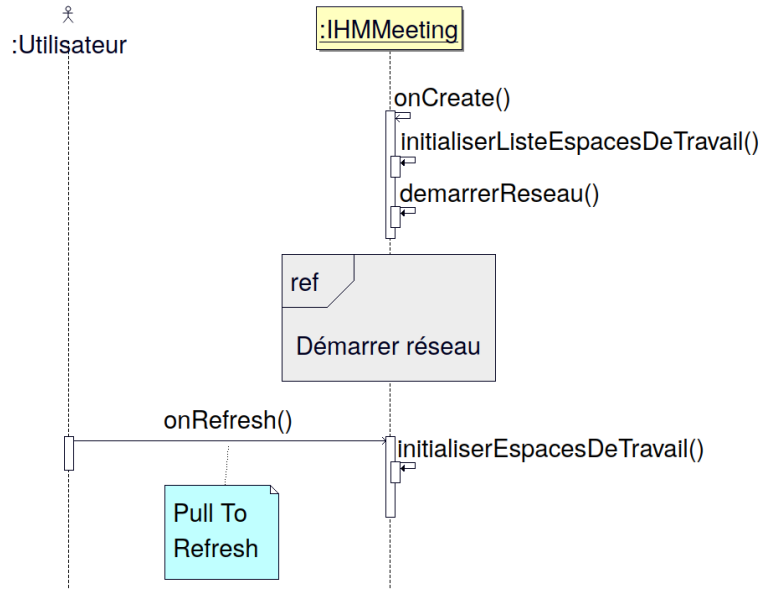
La classe **`Communication`** gère la communication entre l'application mobile et le portier. La classe **`Communication`** est une classe active car elle contient une méthode (**`run()`**) qui s'exécute dans un Thread. Ceci est utile pour assurer la réception par la méthode **`recevoir()`**.



Scénarios

Visualiser la liste des espaces de travail

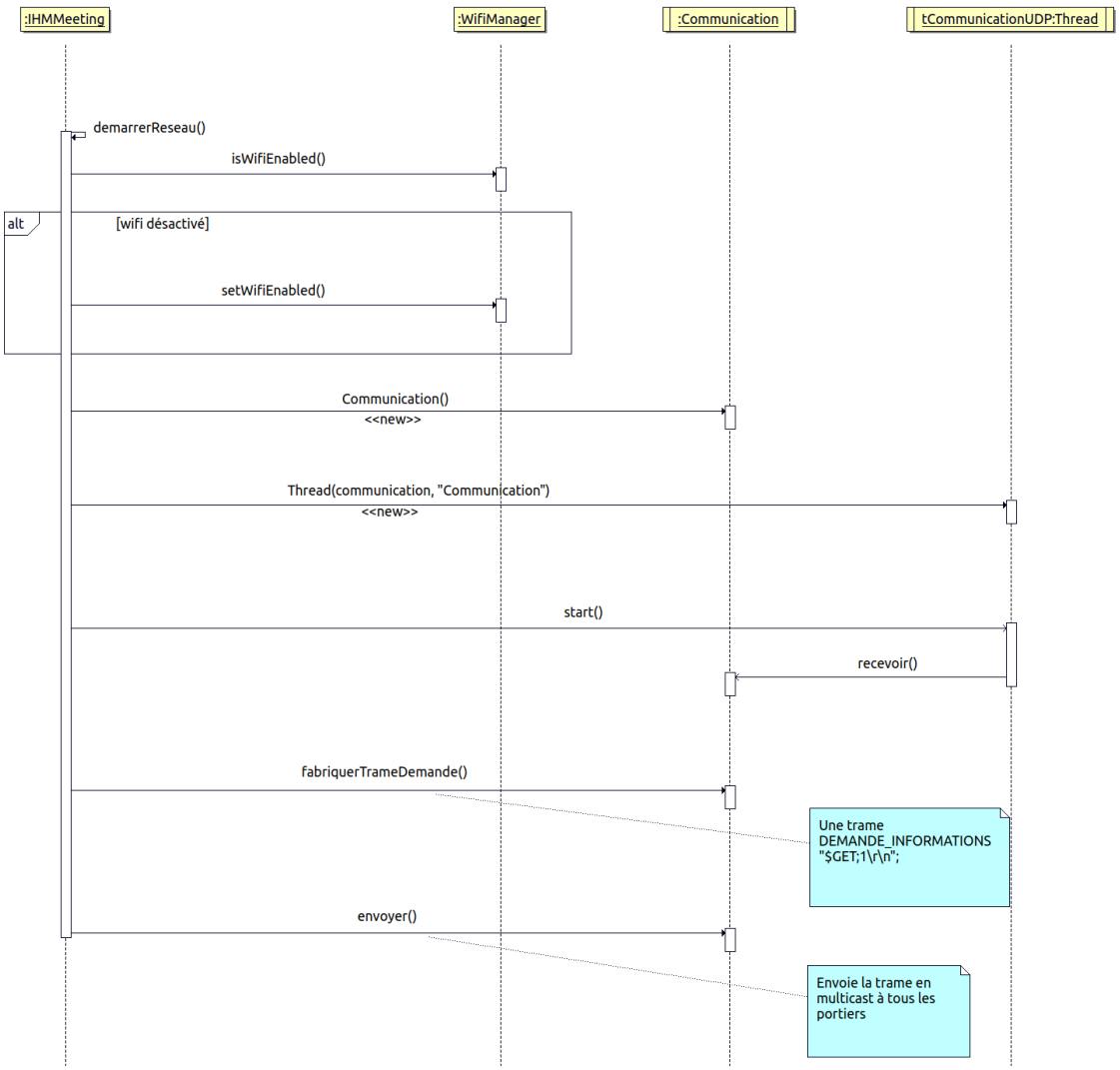
Sur la page d'accueil, l'utilisateur visualise la liste des espaces de travail détectés :



La méthode **`onCreate()`** est appelée pour créer l'activité. Ensuite, la méthode **`initialiserListeEspacesDeTravail()`** est appelée pour créer la liste des espaces de travail, puis la méthode **`demarrerReseau()`** pour démarrer le réseau et la communication.

Lorsque l'utilisateur actualise l'affichage, la méthode **`initialiserEspacesDeTravail()`** est appelée et affiche les espaces de travail dans la liste.

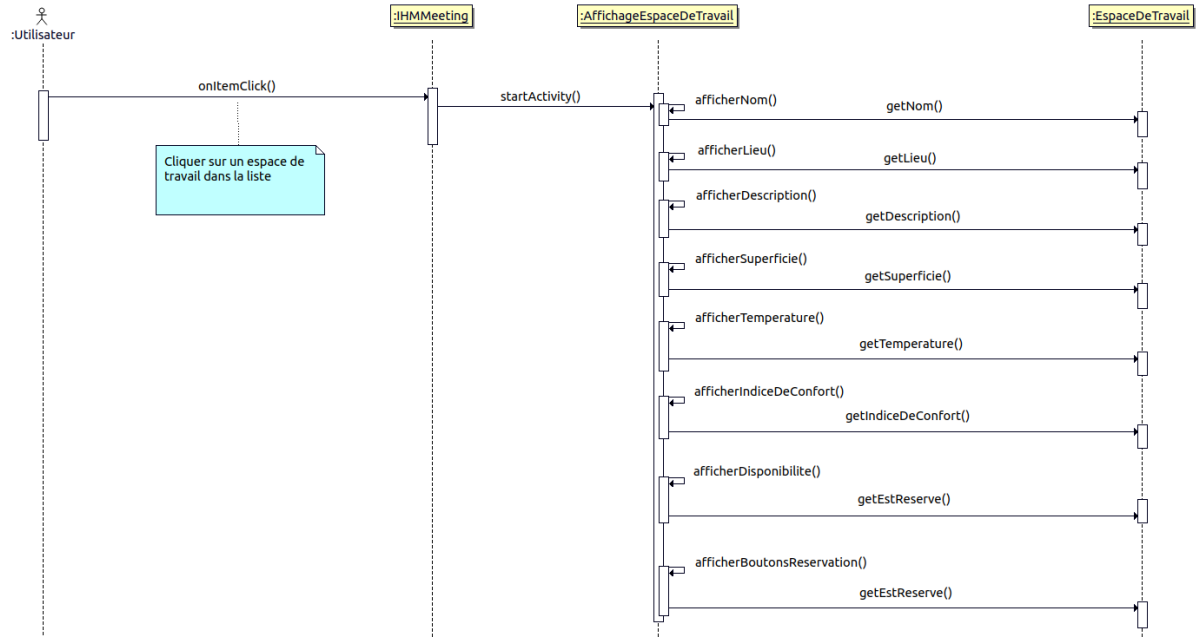
Démarrer le réseau



La méthode **`isWifiEnabled()`** de la classe **`WifiManager`** est appelée pour vérifier si le wifi de l'appareil est activé. Si le wifi est désactivé, la méthode **`setWifiEnabled()`** est appelée pour l'activer.
Les constructeurs **`Communication()`** et **`Thread()`** sont appelés. La méthode **`start()`** est appelée et démarre le thread de la communication.
Les méthodes **`fabriquerTrameDemande()`** et **`envoyer()`** de la classe **`Communication`** sont appelées pour fabriquer une trame de demande d'informations et l'envoyer en multicast à tous les portiers.

Visualiser un espace de travail

Sur la page d'accueil, l'utilisateur visualise l'espace de travail sélectionné :



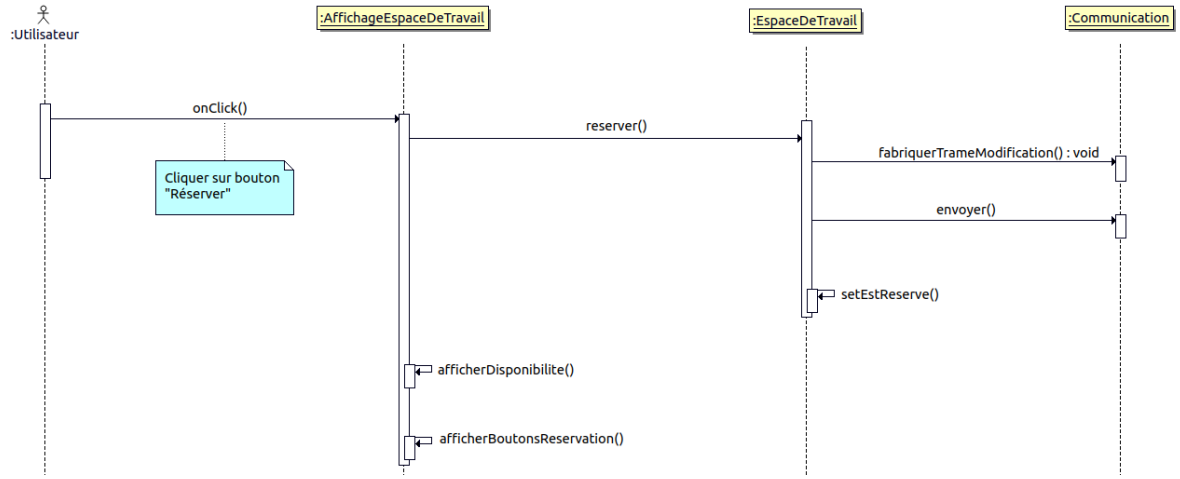
L'utilisateur clique sur un espace de travail dans la liste qui appelle la méthode **`startActivity()`** en passant l'objet **`EspaceDeTravail`**.

```
@Override
public void onItemClick(AdapterView<?> a, View v, int position, long id)
{
   Intent intent = new Intent(IHMMeeting.this,
AffichageEspaceDeTravail.class);
   intent.putExtra("unEspaceDeTravail",
(Serializable)espacesDeTravail.get(position));
   startActivityForResult(intent, 0);
}
```

Lors de la création de l'activité d'affichage de l'espace de travail, les méthodes **`afficherNom()`**, **`afficherLieu()`**, **`afficherDescription()`**, **`afficherSuperficie()`**, **`afficherTemperature()`**, **`afficherIndiceDeConfort()`**, **`afficherDisponibilite()`** et **`afficherBoutonsReservation()`** sont appelées.

Réserver un espace de travail

Sur la page d'affichage de l'espace de travail, l'utilisateur peut "Réserver un espace de travail" :



L'utilisateur clique sur le bouton "Réserver" qui appelle la méthode **`reserver()`** de la classe **`EspaceDeTravail`**.
La méthode **`reserver()`** appelle successivement les méthodes **`fabriquerTrameModification()`** et **`envoyer()`** pour fabriquer et envoyer la trame, puis la méthode **`setEstReserve()`** pour modifier la disponibilité de l'espace de travail (l'attribut **`estReserve`** à "**`true`**").

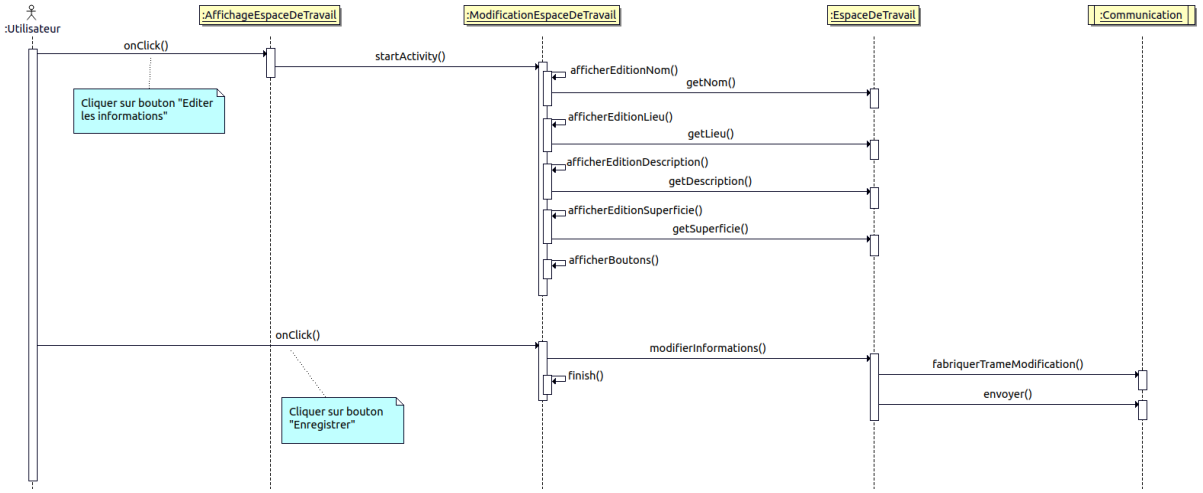
```
public void reserver()
{
   String trame = "\0";
   List<String> parametres = Arrays.asList("0");
   trame =
communication.fabriquerTrameModification(Communication.MODIFICATION_DISP
ONIBILITE, parametres);
   communication.envoyer(trame, adresseIP);

   setEstReserve(true);
}
```

Enfin, les méthodes **`afficherDisponibilite()`** et **`afficherBoutonsReservation()`** de la classe **`AffichageEspaceDeTravail`** sont appelées pour afficher la disponibilité "Occupé" et le bouton "Libérer".

Editer un espace de travail

Sur la page de modification d'un espace de travail, l'utilisateur peut "Editer un espace de travail" :



Sur la page d'affichage d'un espace de travail, lorsque l'utilisateur clique sur le bouton "Editer les informations", la méthode **`startActivity()`** est appelée en passant l'objet **`EspaceDeTravail`**.
Lors de la création de l'activité de modification de l'espace de travail, les méthodes **`afficherEditionNom()`**, **`afficherEditionLieu()`**, **`afficherEditionDescription()`**, **`afficherEditionSuperficie()`** et **`afficherBoutons()`** sont appelées.

Sur la page de modification d'un espace de travail, l'utilisateur clique sur le bouton "Enregistrer" qui appelle la méthode **`modifierInformations()`** de la classe **`EspaceDeTravail`**.

```
public void onClick(View v)
{
   String champs[] = new String[] { editionNom.getText().toString(),
editionDescription.getText().toString(),
editionLieu.getText().toString(), editionSuperficie.getText().toString()
};
   List<String> parametres = Arrays.asList(champs);
   espaceDeTravail.modifierInformations(parametres);

   finish();
}
```

La méthode **`modifierInformations()`** appelle successivement les méthodes **`fabriquerTrameModification()`** et **`envoyer()`** pour fabriquer et envoyer la trame.

```
public void modifierInformations(List<String> parametres)
{
   Log.d(TAG, "modifierInformations()");

   String trame = "\0";
   trame =
communication.fabriquerTrameModification(Communication.MODIFICATION_INFO
RMATIONS, parametres);
   communication.envoyer(trame, adresseIP);
}
```

Tests de validation

Test	Validation OUI / NON
Visualiser la liste des espaces de travail	OUI
Visualiser les informations, la température, l'indice de confort, la disponibilité et la durée d'occupation d'un espace de travail	OUI
Réserver un espace de travail	OUI
Libérer un espace de travail	OUI
Editer les informations d'un espace de travail	OUI
Ajouter un espace de travail aux favoris	OUI
Retirer un espace de travail des favoris	OUI