



# Création d'un nouveau Display

---

Ref : TV310315\_TU\_SM

Rédacteur : Serge Morvan

## Architecture

### Principes

Un Display est un composant, au sens de C<sup>3</sup>, il peut ainsi bénéficier des services du composant GuiAgent par exemple. L'application n'a aucun lien codé avec lui. Le design du nouveau composant est complètement séparé du code. Chaque élément variable possède un id. Le contrôleur, instancié par le Display lors de l'initialisation charge le fichier .fxml. On retrouve dans le code du contrôleur les id de la partie graphique. Préconisations : le composant graphique principal est un Group, son id est : view, chaque Display possède un bouton de fermeture, dont l'id est quit. Le contrôleur hérite de la classe Widget2D qui offre les services de l'interactivité. L'attribut KEY\_NAME, ici : "InstrumentTemplate" permet au Dock de le rechercher et de l'afficher, lorsque l'item associé est sélectionné. Comme ci dessous dans la classe DockManagerImpl du module navisu-app, lors de la création du Dock :

```
instrumentsRadialMenu = RadialMenuBuilder.create()
.centralImage("instrumentsradialmenu150.png")
.createNode(0, "navigation.png",1,"ais.png",1,"template.png",(e)->open("InstrumentTemplate"))
.build();
```

Le nouveau Display devra s'enregistrer auprès du InstrumentDriverManagerServices, dans la classe AppMain du module navisu-launcher.

```
// deploy components
LOGGER.info("\n"
+ componentManager.startApplication(DpAgentImpl.class,
.....
InstrumentTemplateImpl.class
)
);
.....
InstrumentTemplateServices instrumentTemplateServices
    = componentManager.getComponentService(InstrumentTemplateServices.class);
.....
instrumentDriverManagerServices.registerNewDriver(instrumentTemplateServices.getDriver());
```

## Diagramme UML

La figure ci dessous présente un diagramme simplifié pour un Display appelé InstrumentTemplate,

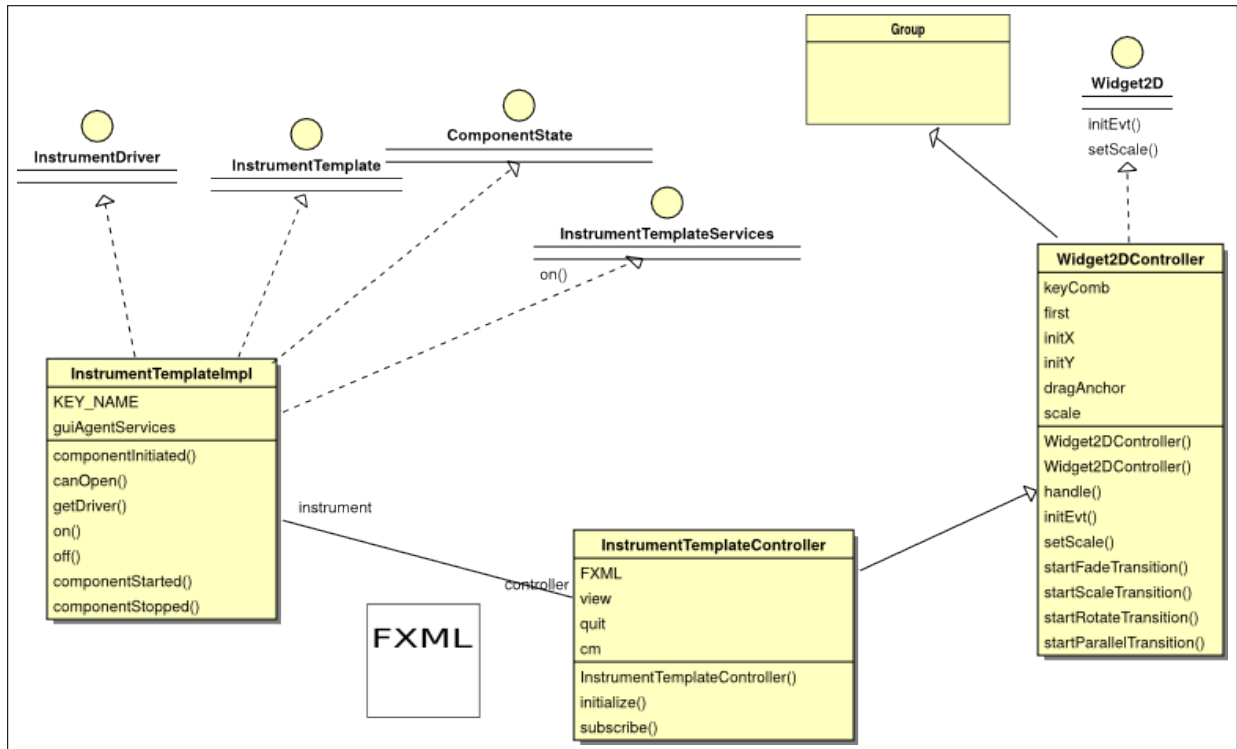


FIGURE 1 – Diagramme UML simplifié

## Créer un nouveau Display : Compass

Reprendre le code du **InstrumentTemplate**, écrire les interfaces **Compass** et **CompassServices**, implémenter les classes **CompassImpl** et **CompassController**. Créer le fichier graphique `compass.fxml`. Identifier par un id chaque variable. Reprendre ces variables en mode public dans le contrôleur. Eventuellement ajouter des services, par défaut un **Display** offre le service `on()`. Implémenter les contrôles.