**Tài liệu thiết kế**

cho dự án

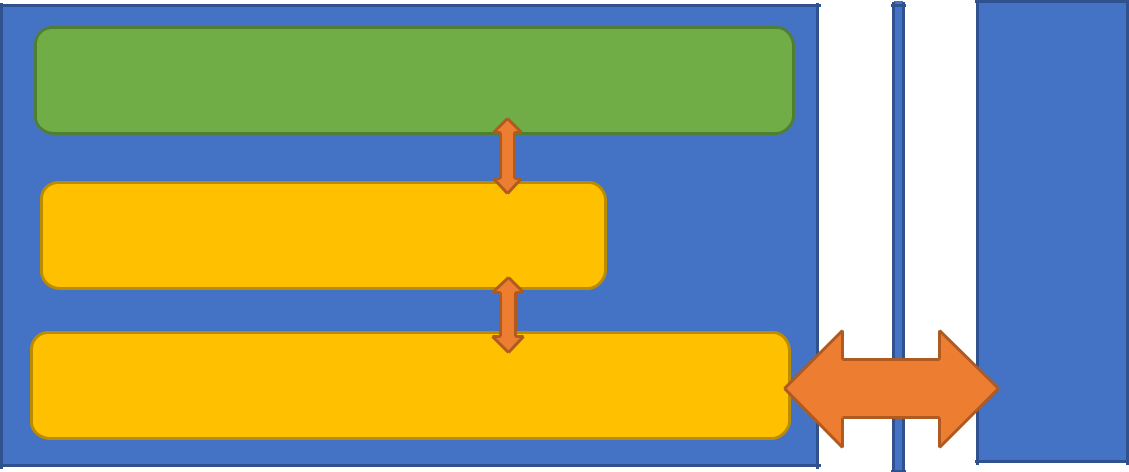
**Home**

Version 1.0

Prepared by TamNT

04/29/2022

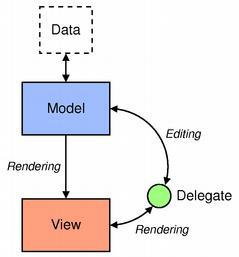
|  |  |  |  |
| --- | --- | --- | --- |
| I. | Thiết kế giao diện |  |  |
|  | Giao diện được trình bày trong Home\_UI\_TamNT.docx | |  |
| II. | Thiết kế tương tác |  |  |
|  | Thiết kế tương tác được trình bày trong Home\_UX\_TamNT.docx | |  |
| III. Thiết kế kiến trúc | |  |  |
| 1. | Sơ đồ tổng thể của ứng dụng |  |  |
|  | View (qml) |  |  |
|  | Model (C++) | Climate |  |
|  | Services |  |
|  |  |  |
|  | Controller (C++) | dbus |  |
|  | View (qml): Đây là nơi quản lý các màn hình, các component được xây dựng | |  |
|  | bằng qml và các resource của việc xây dựng màn hình |  |  |



Model: Là nơi xây dựng dữ liệu cho việc quản lý trạng thái của giao diện từ C++, nó là nơi thể hiện các dữ liệu cho việc xây dựng trạng thái của màn hinh

Controller: Là phần xử lý, điều khiển chương trình, và chịu trách nhiệm kết nối với các services thứ 3 (cụ thể ở đây là climate sevices)

Kiến trúc xây dựng cho trương trình được xây dựng dựa vào kiến trúc Model View

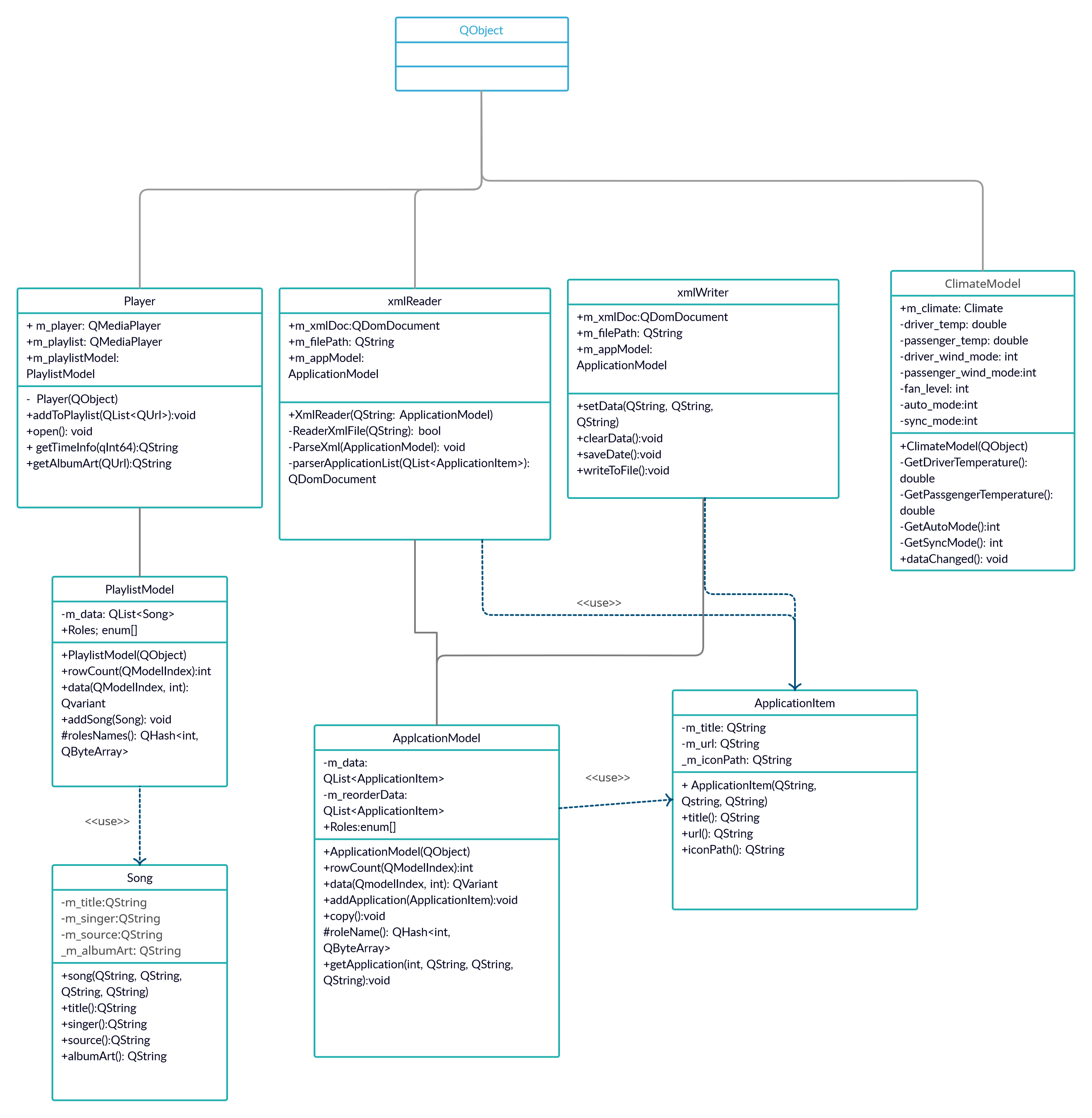


Data: xml chứa thông tin các ứng dụng có trong hệ thống

Model: Class lưu trữ danh sách ứng dụng đọc từ file xml

View: QML hiển thị danh sách ứng dụng

IV. Class Diagram

****

**Description classes:**

**1. Player**

|  |  |  |  |
| --- | --- | --- | --- |
| Function | Descrition | Input | Output |
| Player | Constructor | QObject |  |
| addToPlaylist | Add song to playlist | QList<QUrl> |  |
| open | Open music folder to add  song |  |  |
| getTimeInfo | Format time input to format  mm:ss | qint64 | QString |
| getAlbumArt | Get album art of the song | QUrl | QString |

1. **PlaylistModel**

|  |  |  |  |
| --- | --- | --- | --- |
| Function | Descrition | Input | Output |
| PlaylistModel | Constructor | QObject |  |
| rowCount | Count item in model | QModelIndex | int |
| data | Get roles of model | QModelIndex  Int | QVariant |
| addSong | Add song to model | Song |  |
| roleNames | Set role to model |  | QHash<int, QByte> |

1. **Song**

|  |  |  |  |
| --- | --- | --- | --- |
| Function | Descrition | Input | Output |
| Song | Constructor | QString  QString  QString  QString |  |
| Title  singer | Get title  Get singer |  | QString  QString |
| source | Get source |  | QString |
| Album\_art | Get album art |  | QString |

1. **ApplicationItem**

|  |  |  |  |
| --- | --- | --- | --- |
| Function | Descrition | Input | Output |
| ApplicationItem | Constructor | QString  QString  QString |  |
| title | Get title |  | QString |
| url | Get url |  | QString |
| iconPath | Get path of icon |  | QString |

1. **ApplicationModel**

|  |  |  |  |
| --- | --- | --- | --- |
| Function | Descrition | Input | Output |
| PlaylistModel | Constructor | QObject |  |
| rowCount | Count item in model | QModelIndex | int |
| data | Get roles of model | QModelIndex Int | QVariant |
| addApplication | Add application to model | ApplicationItem |  |
| getAppsList | Get application list after  reorder an application |  | QList<ApplicationItem> |
| copy | Copy default app list to  app list after reorder |  |  |
| roleNames | Set role to model |  | QHash<int, QByte> |
| getApplication | Get application from qml  to add to QList | Int  QString  QString  QString |  |

1. **XmlReader**

|  |  |  |  |
| --- | --- | --- | --- |
| Function | Descrition | Input | Output |
| XmlReader | Constructor | QString  ApplicationModel |  |
| ReadXmlFile | Read XML File | QString | bool |
| PaserXml | Convert data in XML  file to application  model | ApplicationModel |  |
| parserApplicationList | Convert application  model to XML | QList<ApplicationItem> | QDomDocument |

1. **XmlWriter**

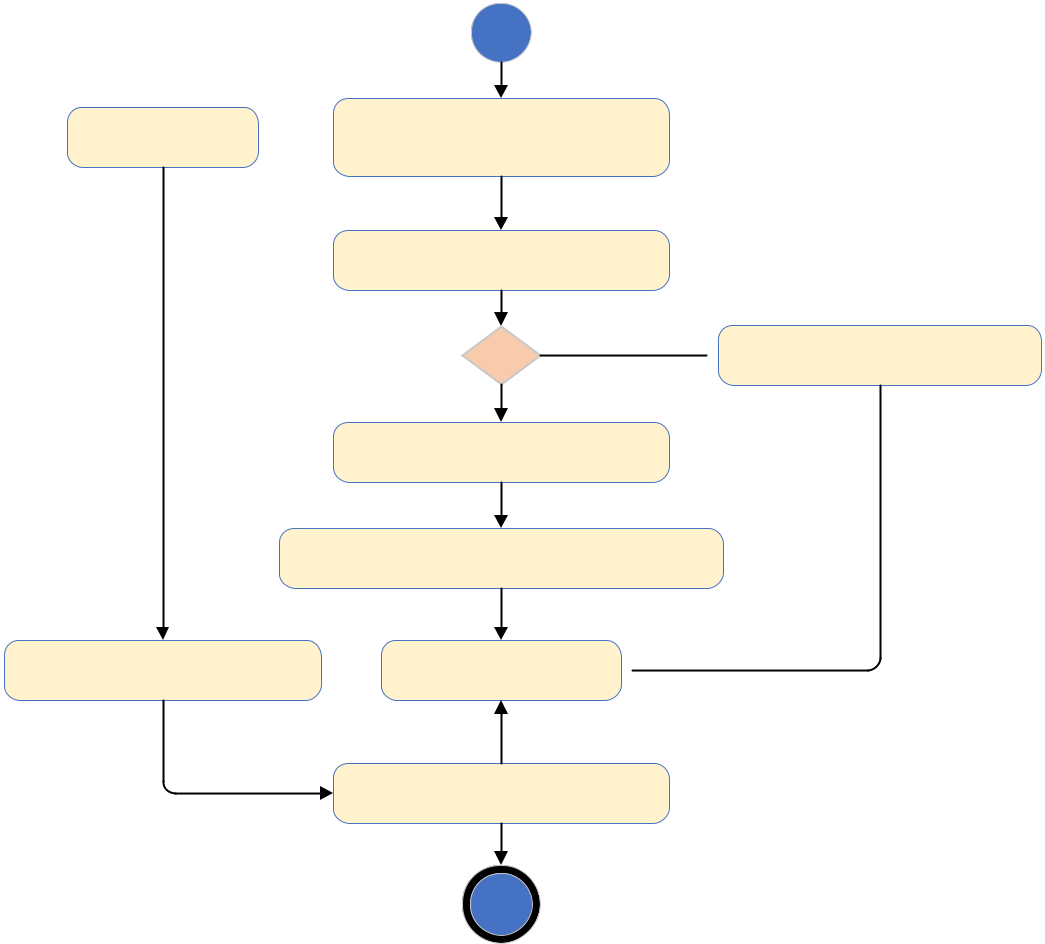
|  |  |  |  |
| --- | --- | --- | --- |
| Function | Descrition | Input | Output |
| XmlWriter | Constructor | QObject |  |
| setData | Set data to each element | QString  QString  QString |  |
| clearData | Clear xml file | QDomDocument |  |
| saveData | Save data from element | QDomDocument | QDomDocument |
| writeToFile | Write data to xml file |  |  |

1. **ClimateModel**

|  |  |  |  |
| --- | --- | --- | --- |
| Function | Descrition | Input | Output |
| ClimateModel | Constructor | QObject |  |
| GetDriverTemperature | Get driver  temperature |  | double |
| GetPassengerTemperature | Get passenger  temperature |  | double |
| GetFanLevel | Get fan level |  | int |
| GetDriverWindMode | Get driver wind mode |  | int |
| GetPassengerWindMode | Get passenger wind  mode |  | int |
| GetAutoMode | Get auto mode |  | int |
| GetSyncMode | Get Sync mode |  | int |

V. Thiết kế luồng xử lý

1. Luồng xử lý khi khởi động ứng dụng Home



Load main qml 

Khởi tạo application

Kiểm tra XML file

[Không có xml]

 Khởi tạo model mẫu

[Có file xml]

Đọc dữ liệu từ xml

Đưa dữ liệu vào QAbtractListModel

Load qml của màn hình  Blinding listmodel 

[Update/request data cho listmodel]

Xử lý user actions

Các bước khởi động chương trình home:

Bước 1: Tạo đối tượng engine của QQmlApplicationEngine

Bước 2: Tạo đối tượng appsModel của ApplicationsModel

Bước 3,4: Tạo đối tượng xmlReader của XmlReader với giá trị truyền vào là đường dẫn đến file xml và đối tượng appsModel

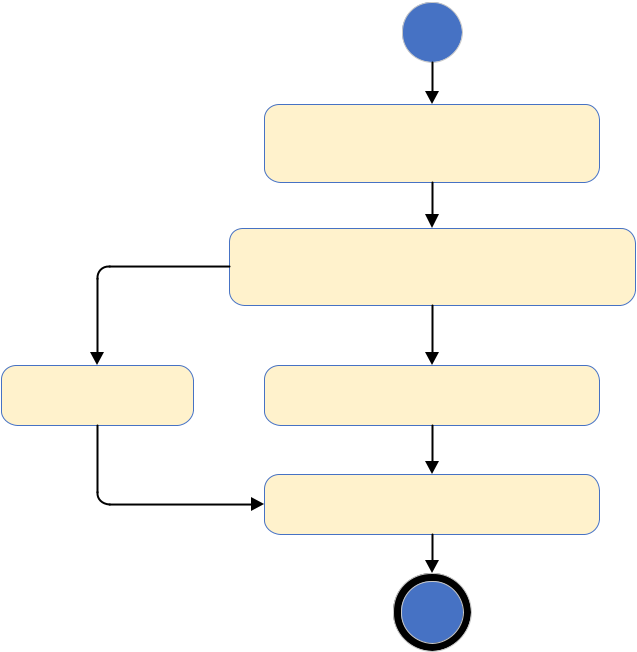
Bước 5: Đọc file xml

Bước 6: Paser thông tin từ xml sang đối tượng ApplicationsModel

Bước 7: Binding appsModel lên QML bằng việc setContextProperty

Bước 8: Khởi động QML engine bằng việc load url của file qml chính

*2. Luồng xử lý khi mở app*



Open App

Đọc dữ liệu từ app listmodel

(Title, Icon, url)

Load qml Load model

Open application

Các bước mở application

Bước 1: Nhận tín hiệu khi khởi động App

Bước 2: Đọc dữ liệu từ application listmodel

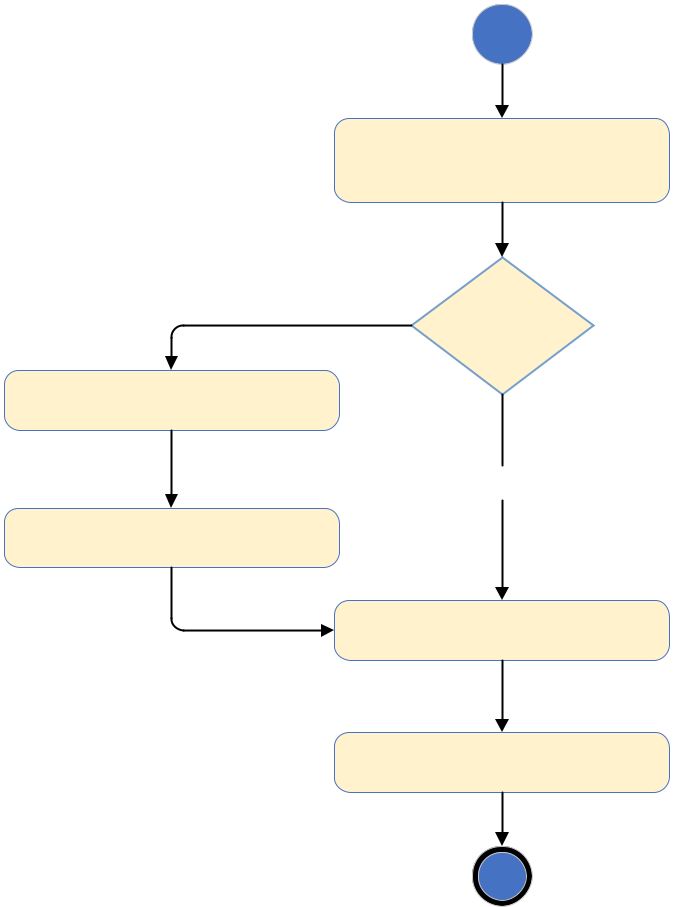
Bước 3: Load model của application tương ứng

Bước 4: Load QML của application

Bước 5: Mở ứng dụng bằng cách Push qml file vào StackView

Bước 6: Thực hiện animator khi ẩn home widget, hiển thị application

3. Luồng xử lý reorder app



Open App

[ClickAndHold] Select

App

change app state: Drag

DropArea: onEnter

Move Application Icon

Display application area

Các bước xử lý khi đổi chỗ app

Bước 1: User khởi động app

Bước 2: Load application list

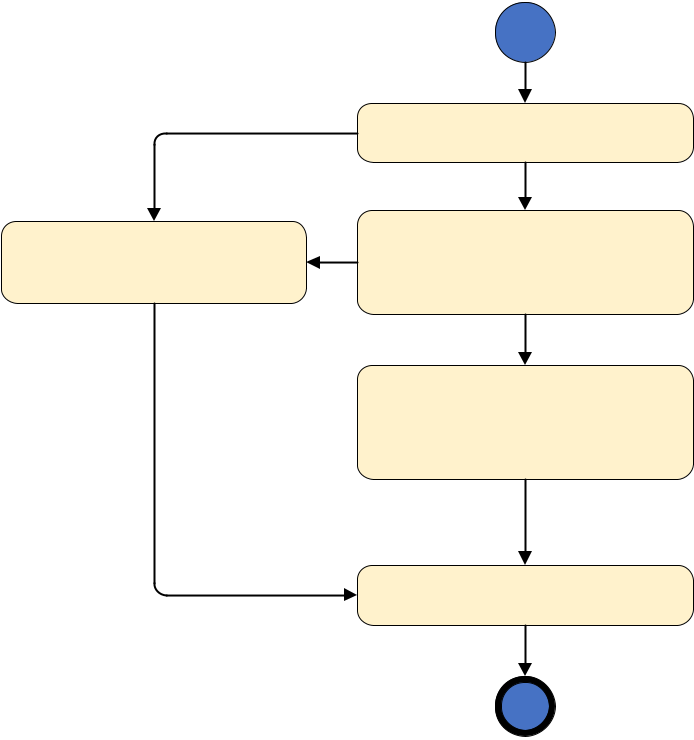
Bước 3:Xử lý onPressAndHold: thay đổi state của app: *app*.state = "Drag"

* Cho phép drag app bằng cách set: Drag.active: *app*.state == "Drag"

Bước 4: Xử lý drop Item trong DropArea và thành công move Application Icon

Bước 5: Hiển thị Application area

4. Luồng xử lý nhận thông tin Widget



Load QML widget

(Media, climate, Map)

Khởi tạo các đối tượng

Đọc thông tin app từ C++

thông qua

setContextProperty

Application change

(media play, next, back...

Climate change

[Update display]

Display Widget

Các bước xử lý khi nhận thông tin widget:

Bước 1: Tạo đối tượng player của Player.

Tạo đối tượng climate của ClimateModel.

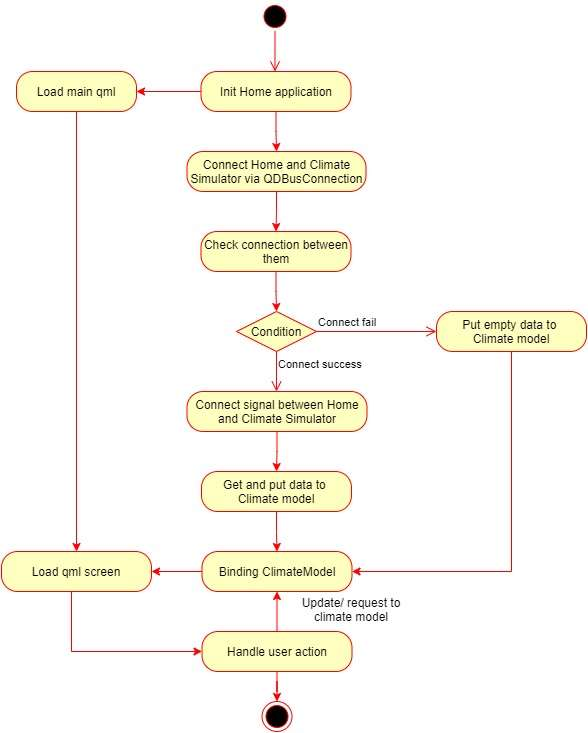
Tạo đối tượng myMap của MyMap.

Bước 2: Blinding các đối tượng lên qml bằng cách setContextProperty.

Bước 3: load các file QML, cụ thể là MapWidget.qml, ClimateWidget.qml và MediaWidget.qml.

Bước 4: Hiển thị các Widget tượng ứng.

1. Luồng xử lý nhận thông tin Climate thông qua DBus



Các bước xử lý khi nhận thông tin DBus:

Bước 1: Tạo đối tượng QqmlApplicationEngine

Tạo đối tượng ClimateModel

Bước 2: Kết nối ClimateModel từ ứng dụng Home với CimateObject qua ứng dụng Simulator thông qua QDBusConnection

Bước 3: Check Connect Fail hoặc Success

Bước 4: Lấy thông tin Climate và đặt vào ClimateModel

Bước 5: Binding ClimateModel đến qml

Bước 6: Hiển thị Climate thông qua Climate widget