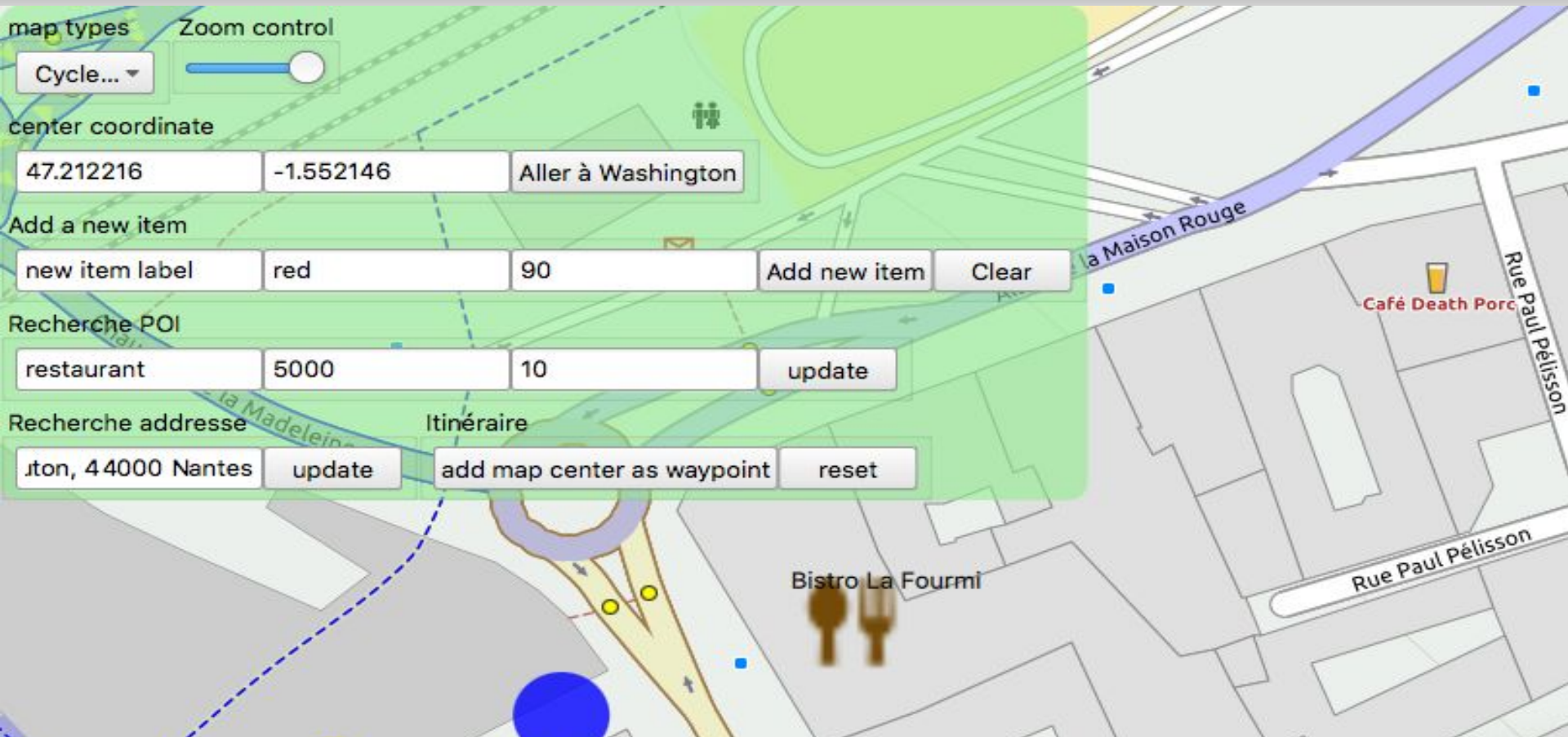


Développement mobile avec

Objectif de l'atelier



Etapes

1. création du projet - (time box : 10mn)
2. ajout de la carte - (time box : 15mn)
3. controler la carte - (time box : 15mn)
4. ajouter des éléments - (time box : 30mn)
5. géocoding - (time box : 5mn)
6. itinéraire - en bonus

Etape 1

création du projet

- Création du projet
- Création de la fenêtre
- Activation des modules
location et positioning

Création du projet

1. Créer un projet de type Application (Qt Quick Application)
2. Modifier le fichier .pro pour activer les modules

```
QT += qml quick positioning location
```

3. Modifier le fichier main.qml pour charger les modules nécessaires

```
import QtPositioning 5.3
import QtLocation 5.5
import QtQuick.Controls 1.4
import QtQuick.XmlListModel 2.0
```

Etape 2

ajout de la carte

- Sélectionner un service de géolocalisation
- Ajouter une carte
- Aller plus loin :
 - Changer de fournisseur de service

```
Plugin {  
  id: myPlugin  
  name: "osm"  
}
```

```
Map{  
  id: myMap  
  anchors.fill: parent  
  plugin: myPlugin  
}
```

```
Plugin{  
  id:mapPlugin  
  name: "mapbox"  
  PluginParameter { name: "useragent"; value: "%VOTRE_APPLI%" }  
  PluginParameter { name: "mapbox.access_token"; value: "%VOTRE_JETON%" }  
  PluginParameter { name: "mapbox.map_id"; value: "mapbox.pencil" } // "wheatpaste/comic/pencil/dark"  
}
```


Etape 3

Controler la carte

- centrer la carte
 - sur une position arbitraire
 - sur la position GPS et/ou réseau
- modifier le zoom
- modifier les gestures
- changer le type d'affichage

Centrer sur la carte sur une position arbitraire ou sur la position du terminal mobile

```
Location{  
  id: myLocation  
  coordinate {  
    latitude : 47.212047;  
    longitude : -1.551647  
  }  
}  
  
Map{  
  id : myMap  
  //...  
  center : myLocation.coordinate  
}
```

```
PositionSource{  
  id : myPos  
  updateInterval : 1000  
  active : true  
  preferredPositioningMethods : PositionSource.AllPositioningMethods  
}  
  
Map{  
  id : myMap  
  //...  
  center : myPos.position.coordinate  
}
```

```
Item{
    id:controlPanel
    width:controls.childrenRect.width
    height:controls.childrenRect.height
    Rectangle{
        anchors.fill: parent
        color:"lightgreen"
        opacity:0.6
        radius:10
    }
    Flow{
        id:controls
        width:parent.width
        //Ajouter les controles ici
    }
}
```

```

Item{ id:controlPanel /* ... */
  Flow{ id:controls /* ... */
    GroupBox{
      title:"map types"
      ComboBox{
        model:myMap.supportedMapTypes
        textRole:"description"
        onCurrentIndexChanged: myMap.activeMapType = myMap.supportedMapTypes[currentIndex]
      }
    }

    GroupBox{
      title:"Zoom control"
      Slider{
        value:myMap.zoomLevel
        minimumValue: myMap.minimumZoomLevel
        maximumValue: myMap.maximumZoomLevel
      }
    }
  }
}

```

```
GroupBox{
  title:"center coordinate"
  Row{

    TextField{
      text:myMap.center.latitude.toFixed(6)
      onEditingFinished: myMap.center.latitude = text
    }
    TextField{
      text:myMap.center.longitude.toFixed(6)
      onEditingFinished: myMap.center.longitude = text
    }
  }
}
```

Etape 4

Ajouter du contenu

- ajouter des objets statiques
 - taille écran
 - taille réelle
- ajouter des objets à partir d'un modèle
- ajouter dynamiquement un nouvel objet
- utiliser un web service comme source de données
- utiliser le service de recherche comme source de données

```
Map{
  id:myMap
  /* ... */
  MapQuickItem {
    coordinate: myMap.center.atDistanceAndAzimuth(50,270)
    sourceItem: Rectangle{
      width:50; height:50; radius:25; color:"blue"; opacity:0.8
    }
  }
  MapCircle {
    center: myMap.center.atDistanceAndAzimuth(100,90)
    opacity:0.8
    color:"red"
    radius:10
  }
}
```

Définition des données...

```
ListModel{  
    id:dummyModel  
    ListElement {  
        Latitude: 47.212047  
        Longitude: -1.551647  
        Label: "something"  
        Orientation: 0  
        Color:"green"  
    }  
    ListElement {  
        /* idem */  
    }  
}
```

...et affichage !

```
Map{  
    id:myMap  
    /* ... */  
    MapItemView{  
        model: dummyModel  
        delegate: MapQuickItem {  
            coordinate: QtPositioning.coordinate(Latitude,  
Longitude)  
            sourceItem: Text{  
                width:100; height:50  
                text:model.Label  
                rotation: model.Orientation  
                opacity: 0.6  
                color:model.Color  
            } }  
        }  
    }  
}
```



```

GroupBox{
    title:"Add a new item"
    Row{
        TextField{ id:newItemName,text:"new item label";}
        TextField{ id:newItemColor,text:"red",;}
        TextField{ id:newItemOrientation,text:"90",;}
        Button{
            text:"Add new item"
            onClicked: {
                dummyModel.append({
                    "Latitude": myMap.center.latitude,"Longitude":myMap.center.longitude,"Label":newItemName.text , "Color":
newItemColor.text, "Orientation":Number(newItemOrientation.text), })
            }
        }
        Button{
            text:"Clear"
            onClicked: {
                dummyModel.clear();
            } } }
    }
}

```

Définition des données...

XmlListModel

```
{
    id: bikeModelXml
    source: "https://www.capitalbikeshare.
com/data/stations/bikeStations.xml"
    query: "/stations/station"
    XmlRole { name: "latitude"; query: "lat/string()"; isKey:
true }
    XmlRole { name: "longitude"; query: "long/string()";
isKey: true }
}
```

...et affichage !

```
Map{
    id: myMap
    /* ... */
    MapItemView{
        model: bikeModelXml
        delegate: MapQuickItem {
            coordinate: QtPositioning.coordinate(model.latitude,
model.longitude)
            sourceItem: Image{ width:50;height:50;source:"qrc:/bike.
png"
        }
    }
}
```

Définition des données...

```
PlaceSearchModel {  
    id: searchModel  
    plugin: myPlugin  
    searchTerm: searchFieldName.text  
    searchArea: QtPositioning.circle(myMap.center, Number  
(searchFieldRadius.text));  
    Component.onCompleted: update()  
    limit: 100  
}
```

les contrôles...

```
GroupBox{  
    title: "Recherche POI"  
    Row{  
        TextField{ id: searchFieldName, text: "restaurant";}  
        TextField{ id: searchFieldRadius, text: "5000";}  
        Button{ text: "update", onClicked: searchModel.update() }  
    }  
}
```

...et affichage !

```
Map{  
    id: myMap  
    /* ... */  
    MapItemView{  
        model: searchModel  
        delegate: MapQuickItem {  
            coordinate: model.place.location.coordinate  
            sourceItem: Image{ width: 64; height: 64;  
                source: model.place.icon.url(Qt.size(64, 64));  
                Text{  
                    anchors.fill: parent  
                    text: model.title  
                }  
            }  
        }  
    }  
}
```

Etape 5

Géocoding

- Transformer une adresse en coordonnée
- Transformer une coordonnée en adresse

Définition des données...

```
GeocodeModel{
  id:geocodeModel
  plugin:myPlugin
  autoUpdate:false
  onLocationsChanged: {
    if (count>0)
      geocodeResult.coordinate = get(0).coordinate;
  }
}
```

...et affichage !

```
MapQuickItem {
  id:geocodeResult
  sourceItem: Image{
    width:50;height:50;
    source:"qrc:/poi.png" }
}
```

les contrôles...

```
GroupBox{
  title:"Recherche adresse"
  Row{
    TextField{ id:searchAddress;text:"11 rue juton, 44000
Nantes";}
    Button{ text:"update"; onClicked: {
      geocodeModel.query = searchAddress.text
      geocodeModel.update()
    }}
  }
}
```

Etape 6

Itinéraire

- paramètres du calcul
- modèle
- contrôler l'itinéraire
- affichage sous forme de liste
- affichage sur la carte

Paramétrage de l'itinéraire

```
RouteQuery{  
  id:routeQuery  
  travelModes:RouteQuery.CarTravel  
  routeOptimizations : RouteQuery.FastestRoute  
}
```

Connexion au model

```
RouteModel {  
  id: routeModel  
  plugin:myPlugin  
  query: routeQuery  
  autoUpdate: false  
}
```

```

ListView {
  id: listview
  anchors.fill: parent
  spacing: 10
  model: routeModel.status == RouteModel.Ready ? routeModel.get(0).segments : null
  visible: model ? true : false
  delegate: Rectangle{
    width: parent.width
    height: 50
    color: "lightgrey"
    Row {
      anchors.fill: parent
      spacing: 10
      property bool hasManeuver : modelData.maneuver && modelData.maneuver.valid
      visible: hasManeuver
      Text { text: (1 + index) + "." }
      Text { text: hasManeuver ? modelData.maneuver.instructionText : "" }
    }
  }
}

```



```
GroupBox{
  title:"Itinéraire"
  Row{
    Button{ text:"add map center as waypoint"; onClicked: {
      routeQuery.addWaypoint(myMap.center);
      routeModel.update();
    }}
    Button{ text:"reset"; onClicked: {
      routeQuery.clearWaypoints();
      routeModel.update();
    }}
  }
}
```



Guillaume Charbonnier
gcharbonnier@a-team.fr