How QML helped me make KStars more interesting

-Samikshan Bairagya

Firstly, what is KStars?

In simple words, it is an application that displays the sky with respect to your location.

...along with many other useful tools

So what did I add to KStars?

"What's interesting..."

Suggest users sky-objects that might interest them.

How interesting a sky-object is, is directly proportional to the visibility of that sky-object in that place and time

What determines visibility?

2 factors:

- Light pollution
- Equipment available to the user

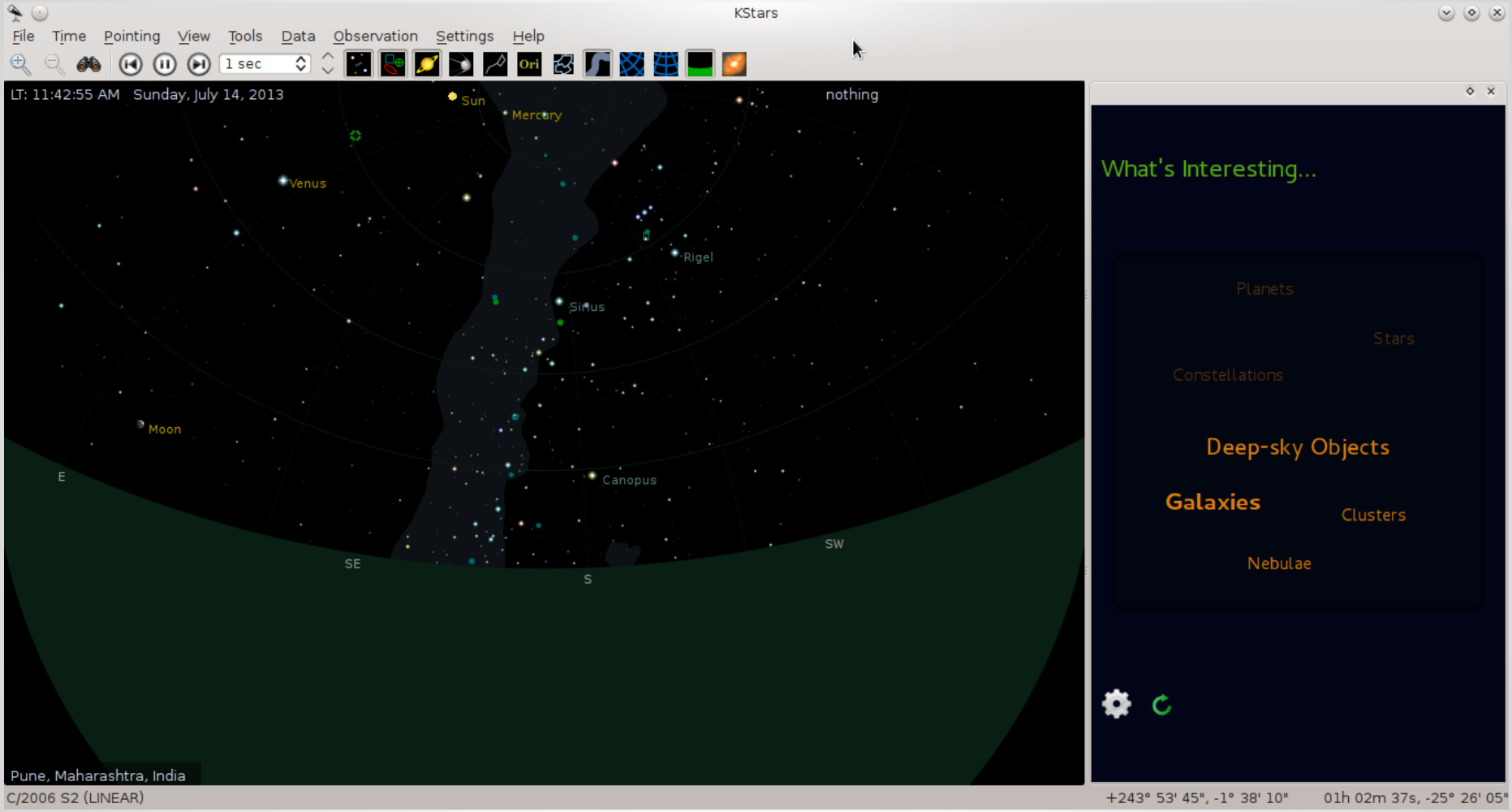
Enough of introduction. Coming to the QML part.

Why QML?

- Looks good
- Lightweight
- Mobile porting of KStars?

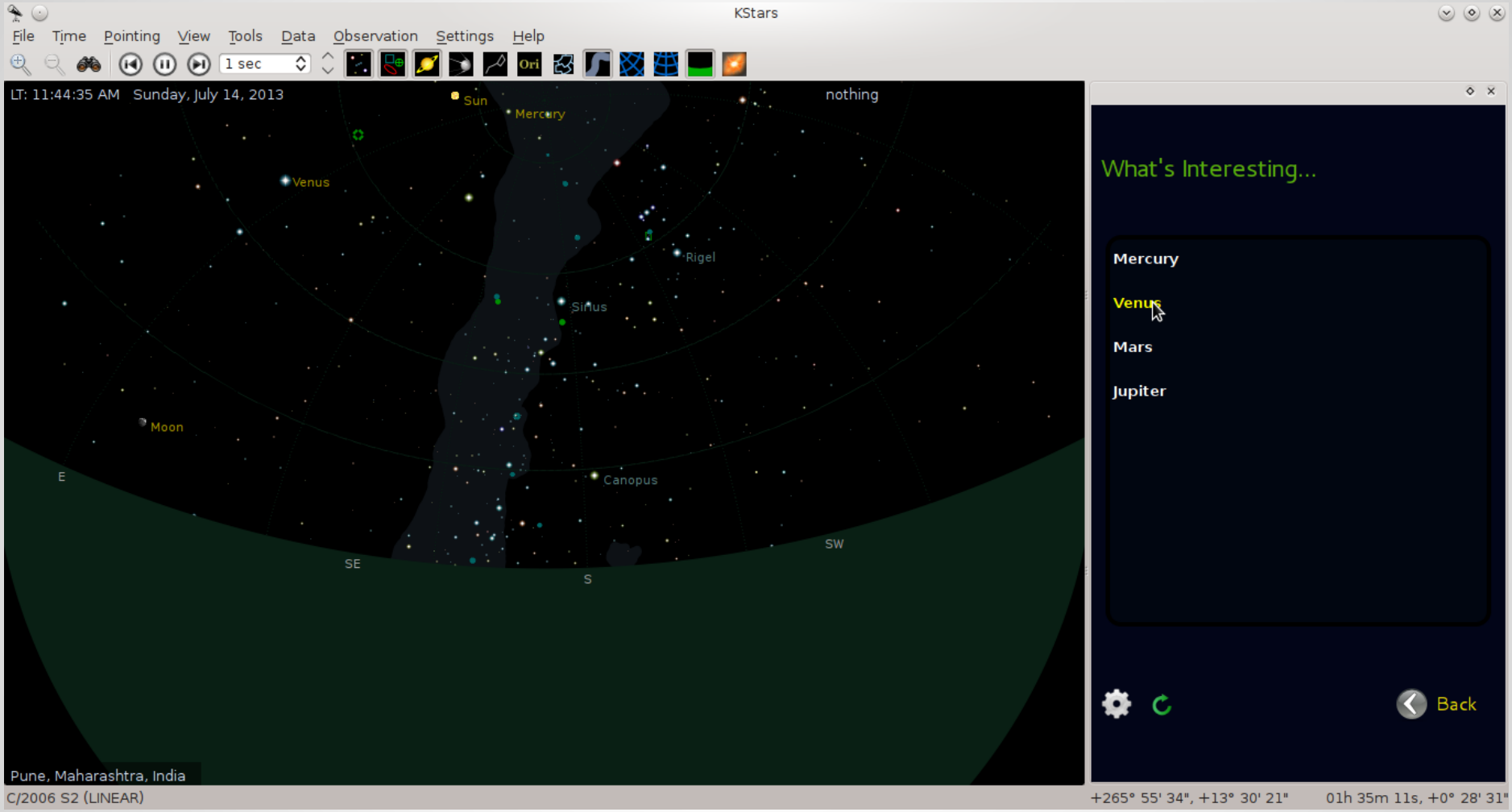
What I needed in the UI

- Area for user to choose between sky-object categories


KStars
File Time Pointing View Tools Data Observation Settings Help
1 sec
LT: 11:42:55 AM Sunday, July 14, 2013
nothing
Sun
Mercury
Venus
Rigel
Sirius
Moon
Canopus
E
SE
S
SW
Pune, Maharashtra, India
What's Interesting...
Planets
Stars
Constellations
Deep-sky Objects
Galaxies
Clusters
Nebulae
C/2006 S2 (LINEAR)
+243° 53' 45", -1° 38' 10"
01h 02m 37s, -25° 26' 05"

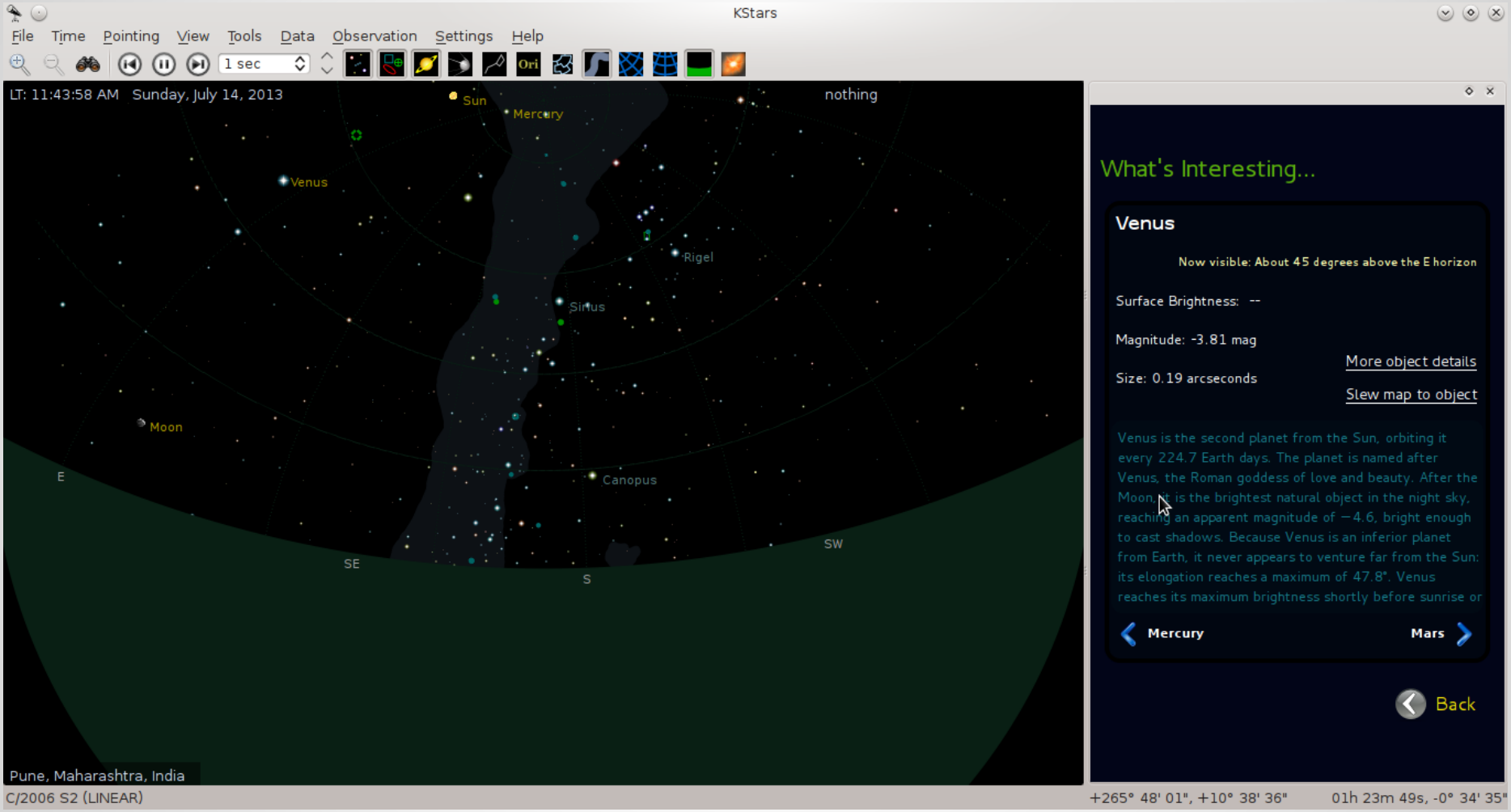
What I needed in the UI

- Area for user to choose between sky-object categories
- View that displays list of visible sky-objects from the category selected.


KStars
File Time Pointing View Tools Data Observation Settings Help
1 sec
LT: 11:44:35 AM Sunday, July 14, 2013
nothing
Sun
Mercury
Venus
Rigel
Sirius
Moon
Canopus
E
SE
S
SW
Pune, Maharashtra, India
What's Interesting...
Mercury
Venus
Mars
Jupiter
Back
C/2006 S2 (LINEAR)
+265° 55' 34", +13° 30' 21"
01h 35m 11s, +0° 28' 31"

What I needed in the UI

- Area for user to choose between sky-object categories
- View that displays list of visible sky-objects from the category selected.
- A details view to display details for a sky-object selected from the above list.


KStars
File Time Pointing View Tools Data Observation Settings Help
1 sec
LT: 11:43:58 AM Sunday, July 14, 2013
nothing
Sun
Mercury
Venus
Rigel
Sirius
Moon
Canopus
E
SE
S
SW
Pune, Maharashtra, India
What's Interesting...
Venus
Now visible: About 45 degrees above the E horizon
Surface Brightness: --
Magnitude: -3.81 mag
Size: 0.19 arcseconds
More object details
Slew map to object
Venus is the second planet from the Sun, orbiting it every 224.7 Earth days. The planet is named after Venus, the Roman goddess of love and beauty. After the Moon, it is the brightest natural object in the night sky, reaching an apparent magnitude of −4.6, bright enough to cast shadows. Because Venus is an inferior planet from Earth, it never appears to venture far from the Sun: its elongation reaches a maximum of 47.8°. Venus reaches its maximum brightness shortly before sunrise or
Mercury
Mars
Back
C/2006 S2 (LINEAR)
+265° 48' 01", +10° 38' 36"
01h 23m 49s, -0° 34' 35"

I need one list-view to rule them all...

Show list of sky-objects of different categories in one list view. The list view in the QML UI should support

- List of interesting planets; or
- List of interesting stars, or
- List of interesting nebulae
- ...
- ...

Model-view to the rescue

QAbstractListModel

SkyObjListModel

```
SkyObjListModel *m_PlanetsModel, m_StarsModel;
m_PlanetsModel = new SkyObjListModel();
m_StarsModel = new SkyObjListModel();
```

SkyObjListModel contains list of SkyObjItems

```
class SkyObjListModel : public QAbstractListModel
{
    Q_OBJECT
    /// Method definitions
private:
    QList<SkyObjItem *> m_SoItemList; //List of sky-object
                                      //items in model
}
```

/// In other words SkyObjItem represents an item in
/// the list of interesting sky-objects

So how to put all this data in the QML List View?

The items of SkyObjListModel is displayed in the QML ListView through a delegate that can access the roles that have been defined for the model.

```
SkyObjListModel::SkyObjListModel(SkyObjItem *soitem,
QObject *parent): QAbstractListModel(parent)
{
    setRoleNames(soitem->roleNames());
}
```

```
class SkyObjItem
{
public:
    enum SkyObjectRoles { DispNameRole = Qt::UserRole + 1,
CategoryRole, CategoryNameRole };
    enum Type { Planet, Star, Constellation, Galaxy, Cluster,
Nebula };
    SkyObjItem(SkyObject *so = 0);
    QVariant data(int role);
    QHash<int, QByteArray> roleNames() const;
    inline QString getName() const { return m_Name; }
    /// More methods
private:
    QString m_Name;        ///Name of sky-object
    QString m_TypeName;    ///Category of sky-object
    QString m_Position;    ///Position of sky-object in the sky.
    Type m_Type; ///Category of sky-object of typeSkyObjItem::Type
    SkyObject* m_So; ///Pointer to SkyObject represented by
                     ///SkyObjItem
};
```

```
QHash<int, QByteArray> SkyObjItem::roleNames() const
{
    QHash<int, QByteArray> roles;
    roles[DispNameRole] = "dispName";
    roles[CategoryRole] = "type";
    roles[CategoryNameRole] = "typeName";
    return roles;
}
```

```
ListView {
    id: soListView
    objectName: "soListObj"
    anchors.fill: parent
    signal soListItemClicked(int type, string typeName, int curIndex)
    ///More code here
    delegate: Item {
        id: soListItem
        x: 5
        height: 40
        Text {
            id: dispText
            objectName: dispName
            text: dispName
            /// Set text font color, size, etc.
            MouseArea {
                /// Define mouse area behaviour here
            }
        }
    }
    model: soListModel     // The model is yet not defined
}
```

How do I define the model for the QML ListView?

Here comes

- QDeclarativeView
- QDeclarativeContext

/* Define a QDeclarativeView object and associate the QML file as source to the QDeclarativeView object */

```
QDeclarativeView m_BaseView = new QDeclarativeView();

m_BaseView->setSource(KStandardDirs::locate("appdata"," tools/whatsinteresting/qml/wiview.qml"));
```

QDeclarativeContext allows passing values to QML components instantiated as QDeclarativeContext objects.

```
QDeclarativeContext m_Ctxt = m_BaseView->rootContext();

/// Bind a C++ model to the soListModel property for the
/// root of the entire context hierarchy. This way the QML
/// ListView can populate itself with items accessed from
/// the C++ model

m_Ctxt->setContextProperty("soListModel",m_ModManager-
>returnModel(type));
```

But,

```
m_Ctxt->setContextProperty("soListModel",m_ModManager->returnModel(type));
```

```
m_ModManager->returnModel(type)) /// What is that???
```

```
/// This returns the corresponding model for the category of
/// sky-object selected by the user
SkyObjListModel* ModelManager::returnModel(int type)
{
    switch(type)
    {
    case 0:  //Planet type
        return m_PlanetsModel;
    case 1:  //Star type
        return m_StarsModel;
    /// More cases
}
```

Signal from QML interface informs which category of sky-object has been selected by the user.

```
/// Inside viewsRow component in the QML file
signal categorySelected(int category)

Rectangle {
    id: planetRect
    // Define properties for rectangle

    Text {
        id: planetText
        //Text properties
        text: i18n("Planets")
        // More properties

        MouseArea {
            id: planetMouseArea
            anchors.fill: parent
            hoverEnabled: true
            onEntered: container.state = "planetAreaEntered"
            onClicked: {
                viewsRow.categorySelected(0)
                container.state = "soTypeSelected"
            }
        }
    }
}
```

The signal is caught by a C++ slot

```
m_ViewsRowObj = m_BaseObj->findChild<QObject *>("viewsRowObj");
connect(m_ViewsRowObj, SIGNAL(categorySelected(int)), this,  SLOT(onCategorySelected
(int)));

void WIView::onCategorySelected(int type)
{
     m_CurCategorySelected = type;
     m_Ctxt->setContextProperty("soListModel", m_ModManager->returnModel(type));
}
```

```
m_BaseView->show();
```

Thank you!

Do checkout KStars and my feature and suggest/implement improvements.

- http://edu.kde.org/kstars/
- git clone git://anongit.kde.org/kstars

References:

1. Light Pollution image: http://en.wikipedia.org/wiki/File: Light_pollution_country_versus_city.png, which was originally posted by Jeremy Stanley here: http://www.flickr.com/photos/79297308@N00/1035660145

2. KDE logo: http://www.kde.org/stuff/clipart/klogo-official-oxygen-128x128.png