1.

QML provides a global JavaScript object called Qt

• Usable anywhere in QML code

• You have seen this in action already with the MouseArea

example:

acceptedButtons: Qt.LeftButton | Qt.RightButton

• All the other enums in the Qt:: C++ namespace are accessible

this way as well!

Also contains a set of functions for

• Creating QML types: Qt.rect(…), Qt.rgba(…), Qt.point(…),

and

• Performing other common operations: Qt.playSound(…),

Qt.openUrlExternally(…), Qt.md5(…)

• Functions for dynamic QML object creation, AJAX and

local database access are provided as well

2.

*Model* classes provide data

• Models can be either in QML (simple cases) or C++ (more

complex cases)

• **QML:** ListModel, XmlListModel, VisualItemModel

• **C++:** QAbstractItemModel, QStringList, QList<QObject\*>

3.

Component {

id: delegate

Item {

id: wrapper; width: 80; height: 80

**scale: PathView.scale**

**opacity: PathView.opacity**

Column {

Image { ... }

Text { ... }

}

}

}

PathView {

anchors.fill: parent; model: MyModel{}; delegate: delegate

path: Path {

startX: 120; startY: 100

**PathAttribute { name: "scale"; value: 1.0 }**

**PathAttribute { name: "opacity"; value: 1.0 }**

PathQuad { x: 120; y: 25; controlX: 260; controlY: 75 }

**PathAttribute { name: "scale"; value: 0.3 }**

**PathAttribute { name: "opacity"; value: 0.5 }**

PathQuad { x: 120; y: 100; controlX: -20; controlY: 75 }

4.

QML provides a global JavaScript object called Qt,it Usable anywhere in QML code, e.g.

Qt.rect(…), Qt.rgba(…), Qt.point(…)

5.