Zurich Python User Group

PySide

Creating GUIs with PySide or PyQt

Aaron Richiger
a.richi@bluewin.ch

5. Dezember 2013

Goals

- What for is the Qt framework?
- What's PySide, PyQt?
- Differences between PySide and PyQt?
- Tools to work with PySide?
- How to create GUIs with PySide?
- User interaction: event handling in Qt?
- What is MVC?
- (Database integration with Qt)?

Qt framework

- cross-platform application and UI framework
- huge C++ library
- Started in 1991 by Trolltech
- Bought by Nokia in 2008
- Given to Open-Source-Community in 2011
- http://qt-project.org/

PySide

- Python bindings for Qt4

```
label = QLabel("Hello World!")
label.setAlignment(Qt.AlignCenter)
```

instead of

```
QLabel *label("Hello world!");
label->setAlignment(Qt::AlignCenter);
```

- open source project (LGPL)

PySide vs. PyQt5

• LGPL	vs.	GNU GPL v3
• open source	vs.	closed source
• Qt5 support planned	vs.	Qt5 support

Both APIs are almost the same, conversion is possible with very little effort.

Getting started

- Install Qt 4.8 (not Qt 5.xy!):

  http://qt-project.org/downloads

- Install PySide:

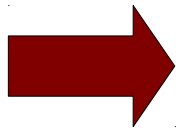
  http://qt-project.org/wiki/PySideDownloads

  Test installation with 'import PySide'

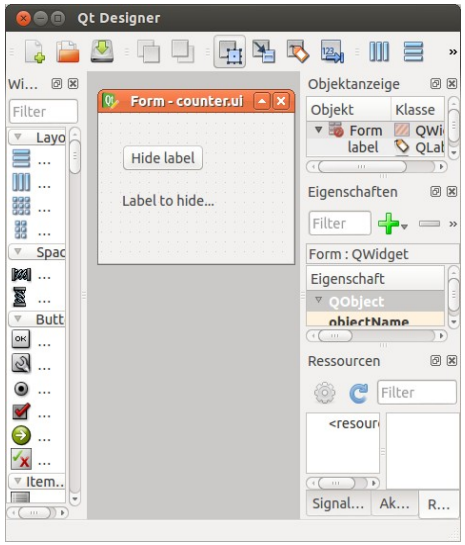
- Text Editor (e.g. Sublime with PySide plugin)
- Qt 4 Designer (WYSIWYG editor)

Example 1: Hello World

```
app = QApplication(sys.argv)
label = QLabel("Hello World")
label.show()
sys.exit(app.exec_())
```



Ex. 2: Three ways to create UIs



Qt 4 Designer

```
Rectangle {
    width: 200
    height: 150
}
```

QML

```
lb = Qlabel(„Hello world“)
lb.resize(300, 400)
```

Programmatically

Example 3: Signals / Slots

- Predefined signals

```
button.clicked.connect(self.on_button)

def on_button():
    print „Button clicked“
```

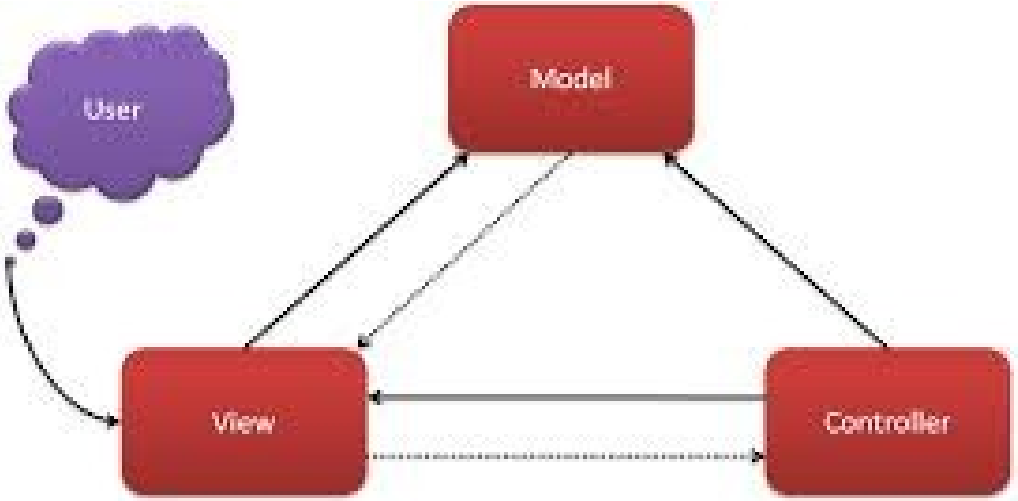
- Custom signals

```
class ClickableLabel(QLabel):
    signal_clicked = Signal()

    def __init__(self):
        ...
        self.signal_clicked.emit()

lb = ClickableLabel()
lb.signal_clicked.connect(lb.hide)
```

Ex. 4: MVC: Model – View - Controller

- General design pattern
- Qt was written with MVC in mind

- Model: Models for (real) objects, e.g. Person
- View: Define the look of the views
- Controller: Manages interaction between Model & View



Example 5: QSql

- For applications with a database
- Qt has drivers for many relational databases
- Special table views
- Resolves foreign key references
- Makes working with databases much simpler

Conclusion

- What for is the Qt framework?
- What's PySide, PyQt?
- Differences between PySide and PyQt?
- Tools to work with PySide?
- How to create GUIs with PySide?
- User interaction: event handling in Qt?
- What is MVC?
- (Database integration with Qt)?