ООП

Семестр 2. Лекция 2

Кафедра ИВТ и ПМ

2018

План

Прошлые темы

Qt

- qDebug. Отладочный вывод
- QObject
- Core classes

Outline

Сигналы и слоты

- Что такое сигнал?

Сигналы и слоты

- Что такое сигнал?
  Сигнал метод вызываемый во время события.
  Что такое слот?

Сигналы и слоты

- Что такое сигнал?
  Сигнал метод вызываемый во время события.
  Что такое слот?
  слот - метод, принимающий сигнал.

Сигналы и слоты

- Как объявляются сигналы и слоты?
- В каком классе можно объявлять сигналы и слоты?

Сигналы и слоты

- Как объявляются сигналы и слоты?
- В каком классе можно объявлять сигналы и слоты?
  В классе построенном на основе QObject
- Какие макросы нужно использовать при описании этого класса?

Сигналы и слоты

- Как объявляются сигналы и слоты?
- В каком классе можно объявлять сигналы и слоты?
  В классе построенном на основе QObject
- Какие макросы нужно использовать при описании этого класса?
  Макрос Q_OBJECT позволяет moc (met object compiler) транслировать код класса на чистый C++.

Сигналы и слоты

```
class A : public QObject{
    Q_OBJECT
...
signals:
    void my_signal();  // реализация не нужна
    };

class B : public QObject{
    Q_OBJECT
...
public slots:
    void my_slot();  // нужно реализовать
    };
```

Сигналы и слоты

- Как соединить сигнал со слотом?

Сигналы и слоты

- Как соединить сигнал со слотом?

```
QObject::connect(button, SIGNAL(clicked(bool)),
                 label, SLOT(clear()) );
```

или

```
QObject::connect(button, &QPushButton::pressed,
                 label, &QLabel::hide);
```

Сигналы и слоты

- Как организовать вызов сигнала и слота с одинаковым фактическим параметром?

Сигналы и слоты

- Как организовать вызов сигнала и слота с одинаковым фактическим параметром?
  Фактический параметр будет одинаковый для сигнала и слота если у них одинаковы формальные параметры.

```
QObject::connect(spinBox, &QSpinBox::valueChanged,
                 label,   &QLabel::setNum);
```

Outline

Outline

Отладочный вывод

В целях изучения Qt полезно использовать так называемый отладочный вывод.
Информация будет напечатана в консоль. Для GUI приложений вывод отладчика можно просмотреть в Qt Creator.

- Для отладочного вывода используется объект класса QDebug.
- Этот объект объявлен в модуле QDebug
- Чтобы получить к нему доступ используется функция qDebug()
- Для вывода используется оператор «

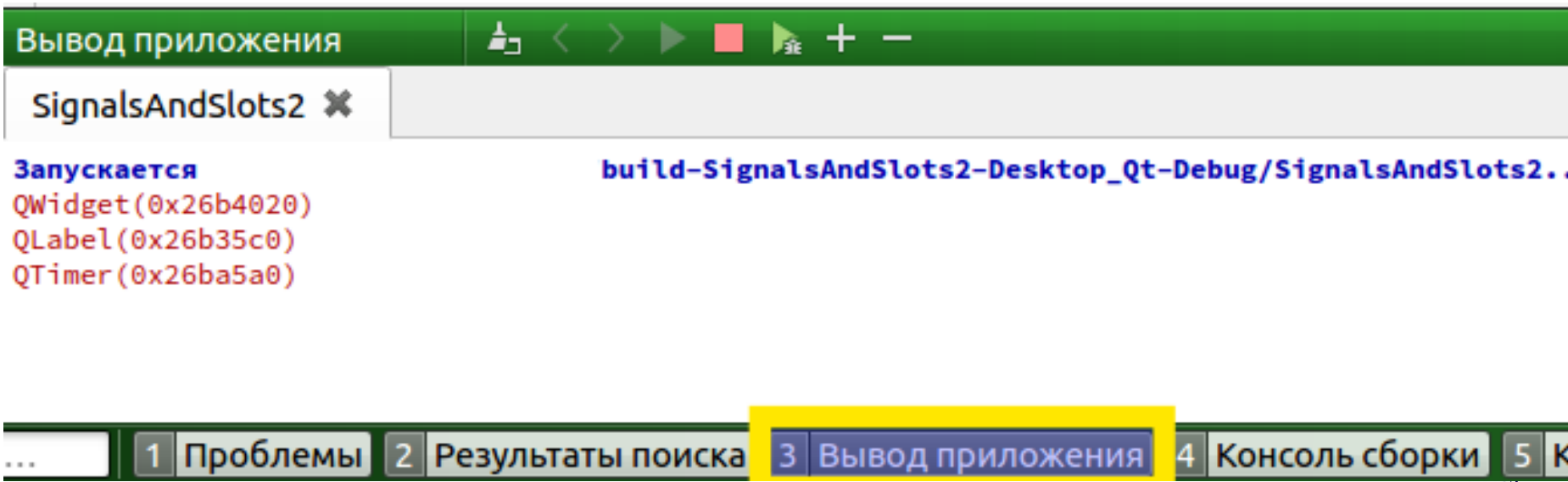
Отладочный вывод

Преимущество вывода через QDebug перд cout:

- QDebug может выводить все объекты унаследованные от QObject
- QDebug может выводить на экран тип объекта при задействованном позднем связывании.

Отладочный вывод

```
QObject * o = new QWidget();
qDebug() << o; // должен быть использован указатель
o = new QLabel();
qDebug() << o;
o = new QTimer();
qDebug() << o;
```



Outline

```
QT += core
```

```
##include <QObject>
```

QObject - базовый виртуальный класс для всех остальных классов в Qt.
Любой класс построенный на основе QObject должен содержать макрос Q_OBJECT. :

connect - соединения сигнала и слота

disconnect - разрыв связи между сигналом и слотом

QObject

Конструктор

```
QObject(QObject *parent = Q_NULLPTR)
```

В конструкторе QObject можно указать ссылку на владельца данного объекта.
При уничтожении владельца, автоматически будут вызваны деструкторы всех его дочерних[1] объектов.

Это даёт возможность не заботится о уничтожении многих объектов создаваемых динамически.
Например владельцем всех элементов интерфейса будет класс главного окна.

[1] под дочерними объектами понимаются не наследники, а агрегированные объекты, т.е. объекты время жизни которых зависит от времени жизни родительского объекта

QObject

- `const QObjectList &QObject::children() const`
- `QObject *QObject::parent() const`

Так как все классы строятся на основе виртуального QObject то тип возвращаемых элементов будет определён во время выполнения программы.

О документации

В документации в описание каждого класса включены свойства (**properties**). Это в основном закрытые поля класса, доступ к которым возможен только с помощью методов.

Как правило методы возвращающие значения названы так же как и свойства, а методы устанавливающие значения начинаются с префикса get.

objectName

Свойство objectName содержит строку (QString) - имя *объекта*.

Это свойство можно использовать чтобы найти определённый объект по имени.
По умолчанию объект имеет пустое имя.

Методы доступа:

```
QString          objectName() const
void          setObjectName(const QString &name)
```

QMetaObject

В Qt каждый класс содержит метаинформацию о самом себе.

Эта метаинформация содержится в классе QMetaObject. Она включает:

- className() - имя класса
- информацию о классе предоставленную разработчиком с помощью макроса Q_CLASSINFO
- Информацию о методах (их количество, названия, ...)

Q_CLASSINFO

Данная метаинформация о классе представлена в виде пар имя-значение.
В примере приведена информация об авторе класса и ссылка на сайт.

```
class MyClass : public QObject
{
    Q_OBJECT
    Q_CLASSINFO("Author", "Pierre Gendron")
    Q_CLASSINFO("URL", "http://www.my-organization.qc.ca")

public:
    ...
};
```

QMetaObject

```
QLabel label("Hello World!");

qDebug() << label.metaObject()->className();
qDebug() << label.metaObject()->methodCount();
```

Вывод:

QLabel
44

Outline

Core classes

Некоторые из core classes
QSize QRect QPoint
QString QVector QStringList QStack QSet QPair QMap QList
QTime QDate QTimer
QException
QRegExp
QEvent
QUrl
QTextStream QFile
QMessageLogger

Ссылки и литература

1. Г. Буч. Объектно-ориентированный анализ и проектирование с примерами приложений. 720 с. 2010 г. 700 страниц. Теория. Примеры на C++. Картинки! Вторая половина книги - примеры OOA и OOD с UML диаграммами.
2. MSDN - Microsoft Developer Network
3. Qt 5.X. Профессиональное программирование на C++. Макс Шлее. 2015 и более поздние издания г. 928 с. Книга периодически обновляется с выходом новых версий фреймворка Qt.
4. www.stackowerflow.com - система вопросов и ответов
5. draw.io — создание диаграмм.

Материалы курса

Слайды, вопросы к экзамену, задания, примеры

github.com/VetrovSV/OOP