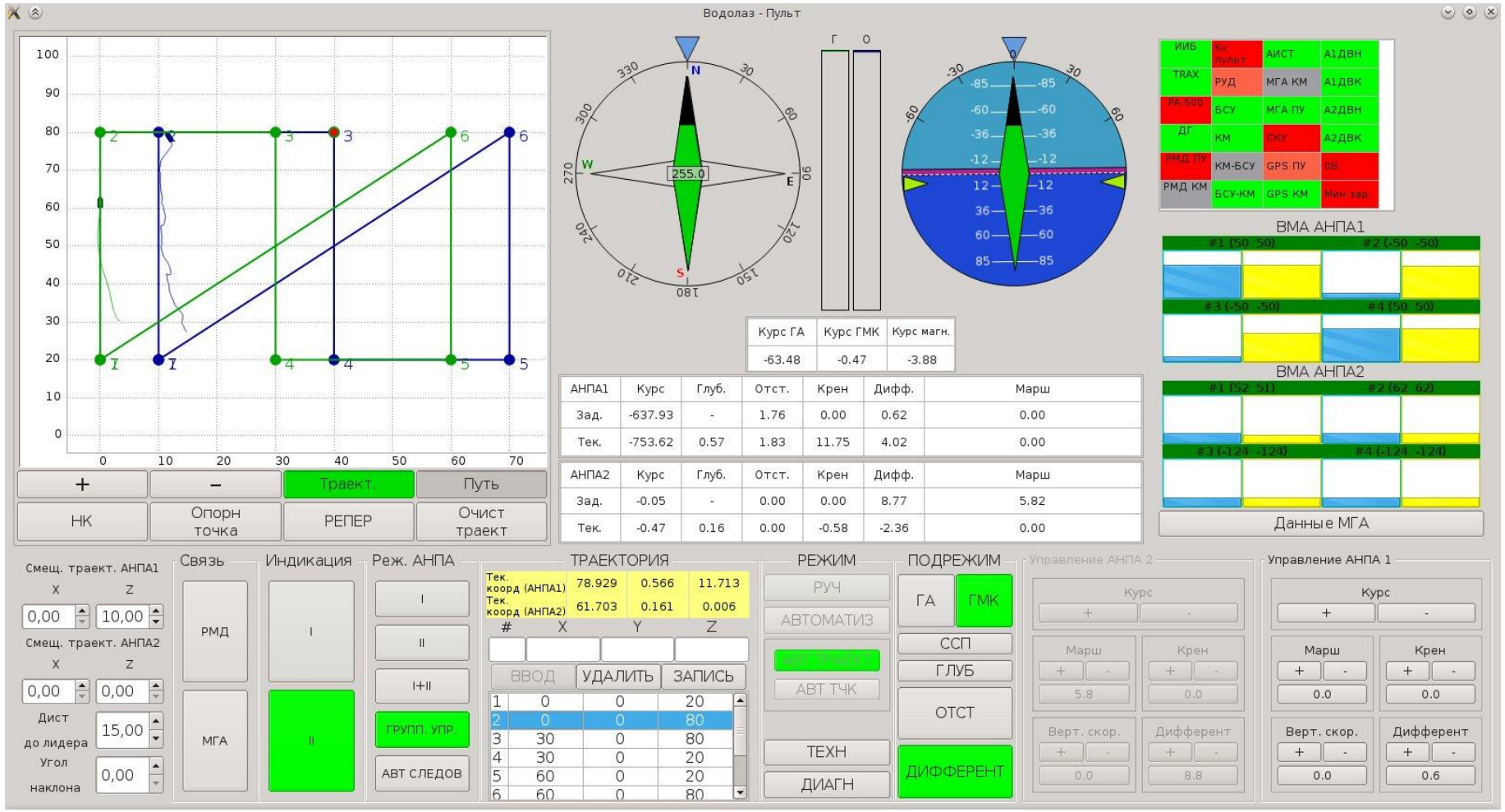
СЕМИНАР 3

Сигналы и слоты

Сигналы и слоты. Введение.



Особенности Qt. Механизм «сигнал-слот».

- Механизм «сигнал-слот» используется для коммуникации между объектами.

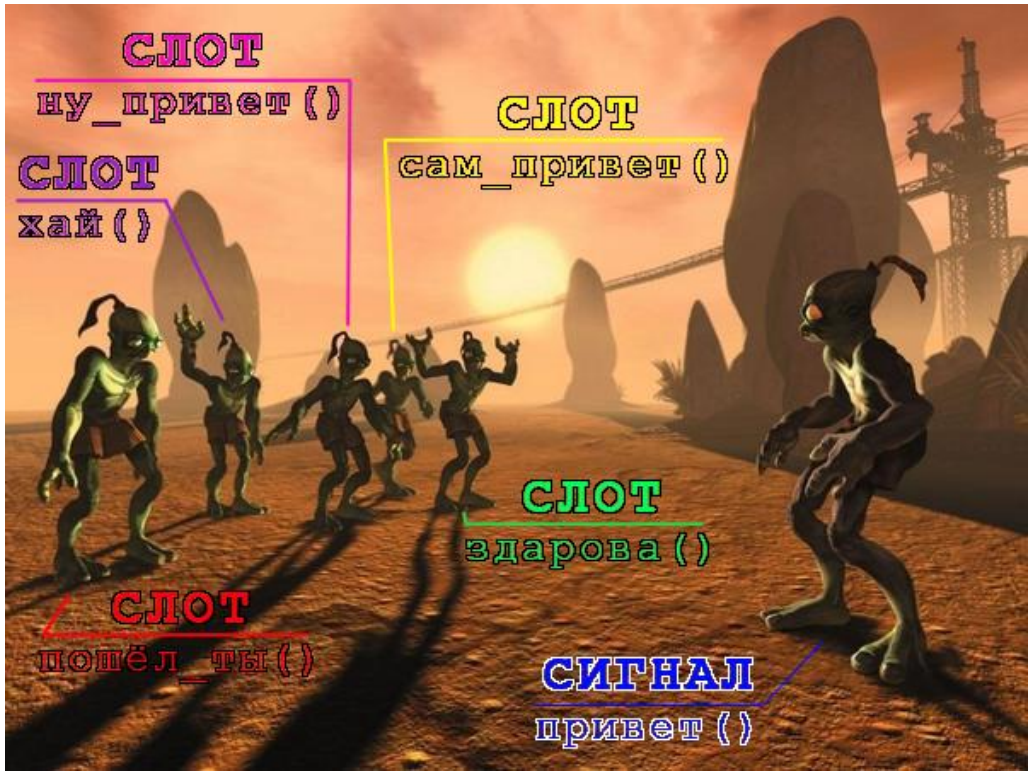


Рис. Механизм «сигнал-слот» в исполнении мудакенов (c) wiki ИУ5

Сигнал – метод, который в состоянии осуществить пересылку сообщений.

Слот – метод, который присоединяется к сигналам (вызывается в ответ на определенный сигнал).

Особенности Qt. Механизм «сигнал-слот».

Преимущества, которые дает программисту механизм «сигнал-слот»:

1. Соединяемые сигналы и слоты абсолютно независимы и реализованы отдельно друг от друга:
    1. Упрощается декомпозиция большого проекта и параллельная разработка;
    2. Предоставлен гибкий механизм для проектирования.
2. Соединение сигналов и слотов можно производить в любой точке приложения.

Особенности Qt. Механизм «сигнал-слот».

Преимущества, которые дает программисту механизм «сигнал-слот»:

3. Соединение сигнала и слота можно осуществлять даже между объектами, которые находятся в различных потоках.
4. При уничтожении объекта происходит автоматическое разъединение всех сигнально-слотовых связей.
5. Механизм сигналов и слотов типобезопасный

Особенности Qt. Механизм «сигнал-слот».

Одному сигналу может соответствовать много слотов.

Одному слоту может соответствовать много сигналов.

Особенности Qt. Механизм «сигнал-слот».

- Можно соединять сигнал и сигнал!

Особенности Qt. Механизм «сигнал-слот».

Недостатки, связанные с применением механизма «сигнал-слот»:

- Сигналы и слоты не являются частью языка C++, поэтому требуется запуск MOC перед компиляцией программы.
- Данный механизм реализован в классе QObject.
  - Для использования сигналов и слотов, класс должен быть унаследован от Qobject.
  - Класс – наследник QObject не может быть шаблонным.
  - В случае множественного наследования Qobject должен быть первым.
- Для реализации механизма MOC требует записи макроса Q_OBJECT сразу при объявлении класса.
  - Нельзя использовать макрос Q_OBJECT с шаблонными классами.

Особенности Qt. Механизм «сигнал-слот».

Недостатки, связанные с применением механизма «сигнал-слот»:

- Сигналы и слоты медленнее чем механизм callback-ов;
- При некоторых способах формирования сигналов и слотов в процессе компиляции не производится никаких проверок: имеется ли сигнал или слот в соответствующих классах или нет; совместимы ли сигнал и слот друг с другом и могут ли они быть соединены вместе. (хотя в Qt5 реализован новый механизм, который частично устраняет этот недостаток).

Сигналы и слоты. Что такое сигнал?

Сигнал - метод, который в состоянии осуществить пересылку сообщений.

Сигнал объявляется однажды и на этом всё, ему не нужна реализация! (Реализацию сигнала осуществляет MOC за вас)

То, что пишете вы в своих исходниках (pultwidget.h):

pult_sem4/pultwidget.h | marshValueChanged(): void

```
1  #ifndef PULTWIDGET_H
2  #define PULTWIDGET_H
3
4  #include "ui_pultwidget.h"
5
6  class PultWidget : public QWidget, private Ui::PultWidget
7  {
8      Q_OBJECT
9
10 public:
11     explicit PultWidget(QWidget *parent = 0);
12     //сигнал объявляется после ключевого слова signals:
13 signals:
14     //метод сигнала не возвращает никаких значений
15     //так что перед названием метода долэен быть указан тип void
16     void marshValueChanged();
17 };
18
19 #endif // PULTWIDGET_H
20
```

Сигналы и слоты. Что такое сигнал?

То, что дописывает MOC в (moc_pultwidget.cpp) за вас:

```
124  // SIGNAL 0
125  void PultWidget::marshValueChanged()
126  {
127      QMetaObject::activate(this, &staticMetaObject, 0, Q_NULLPTR);
128  }
```

Сигналы и слоты. Как выслать сигнал?

Для вызова сигнала используется ключевое слово `emit`.

```
11  //данная конструкция приводит к вызову метода marshValueChanged()
12  //обработка сигнала слотом будет происходить в зависимости от способа соединения
13  void PultWidget::sendMarshSignal() {
14      emit marshValueChanged();
15  }
16
```

Сигналы могут высылаться только объектами классов, которые их содержат.

*В нашем примере только объектами класса `PultWidget`.

Кроме того, сигналы могут высылать информацию, передаваемую в параметре.

Сигналы и слоты. Что такое слот?

Слоты – методы, которые присоединяется к сигналам.

По сути обычные методы, но отличие в том, что могут принимать сигналы!

Объявляются в классе после ключевых слов

`private slots:`

`protected slots:`

`public slots:`

pultwidget.h:

```
class PultWidget : public QWidget, private Ui::PultWidget
{
    Q_OBJECT

public:
    explicit PultWidget(QWidget *parent = 0);
    //сигнал объявляется после ключевого слова signals:
signals:
    //[1] метод сигнала не возвращает никаких значений
    //так что перед названием метода долэен быть указан тип void
    void marshValueChanged();
public:
    //тестовый метод, который генерит сигнал marshValueChanged()
    void sendMarshSignal();

    //[2]. Слот объявляется после ключевого слова public slots:
public slots:
    void addMarshValue();
    //[3]. Создадим переменные, для хранения заданных значений и
    //величин приращения значений
private:
    //переменная, которая хранит заданное значение по маршу
    double marshValue;
    //переменная, которая хранит значение шага изменения марша
    double deltaMarshValue;

```

Сигналы и слоты. Что такое слот?

Реализация функции слота в pultwidget.cpp:

```cpp
#include "pultwidget.h"

PultWidget::PultWidget(QWidget *parent) :
    QWidget(parent)
{
    setupUi(this);
    //проинициализируем значения
    marshValue = 0;
    deltaMarshValue =0.1;
}

void PultWidget::addMarshValue()
{
    //при нажатии на кнопку "+" увеличиваем значение  marshValue на deltaMarshValue
    marshValue+=deltaMarshValue;
    //выведем текущее заданное значение на кнопке btnResetMarsh
    btnResetMarsh->setText(QString::number(marshValue));
}
```

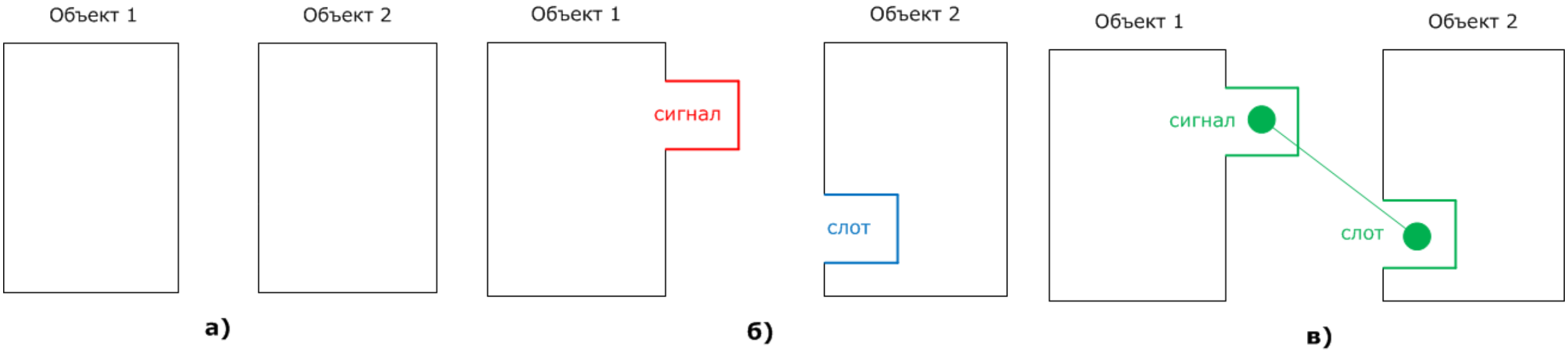
*классы Qt содержат множество уже реализованных слотов.

Сигналы и слоты. Как соединить сигнал и слот?

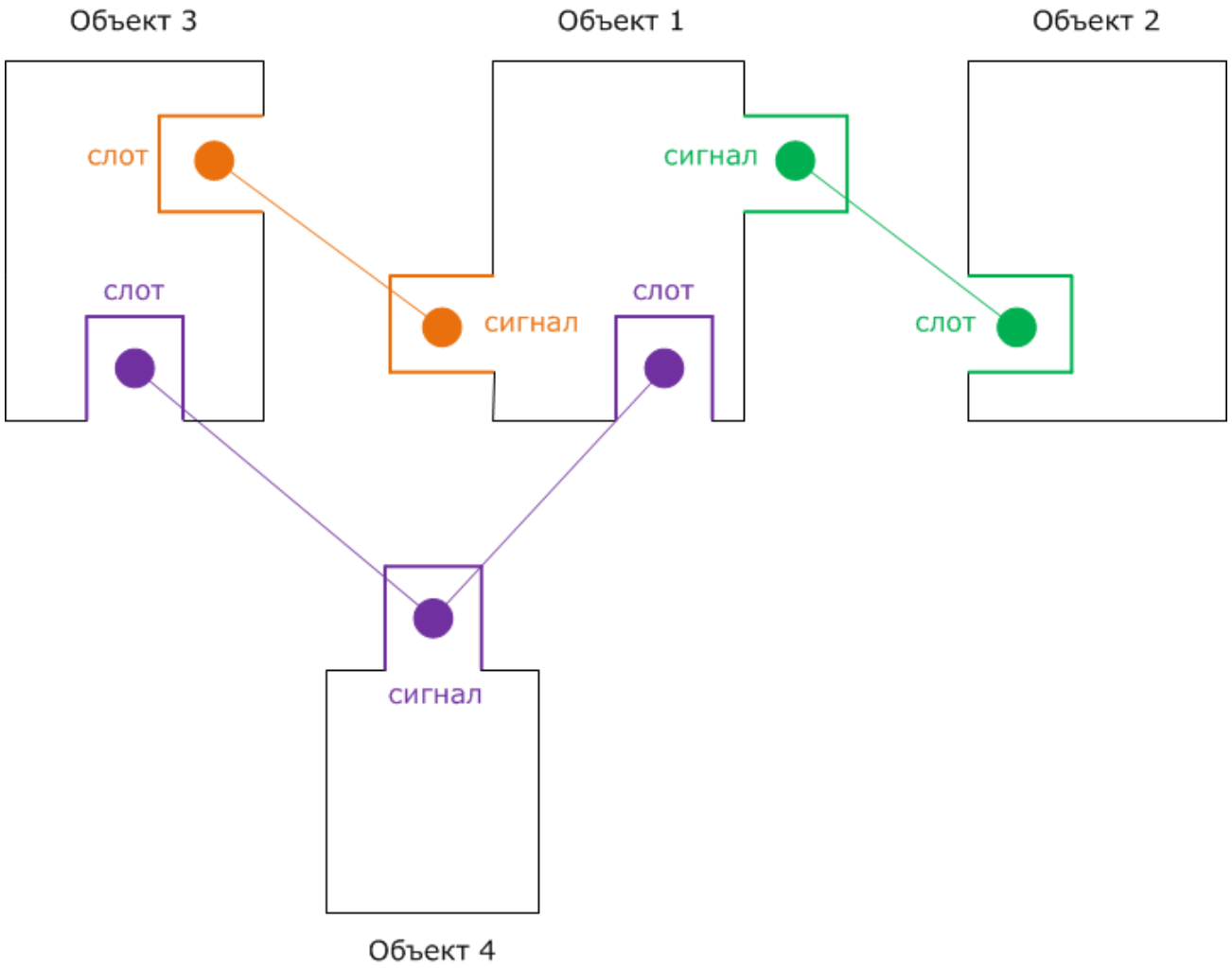
Связь между объектами устанавливается следующим образом: у одного объекта должен быть сигнал, а у второго – слот.

Чтобы соединить два объекта, нужно:

- создать у одного сигнал, а у второго слот;
- соединить сигнал первого и слот второго.



Сигналы и слоты. Как соединить сигнал и слот?



Сигналы и слоты. Как соединить сигнал и слот?

1. Метод connect():
    1. С использованием макросов SIGNAL() и SLOT();
    2. С использованием указателей на функции;
    3. Соединение сигналов с лямбдами.
2. Автоматическое соединение сигналов и слотов.

Практическая часть 1.

Соединить сигналы и слоты таким образом, чтобы:

1. При нажатии на кнопку «+» заданная скорость, отображаемая на кнопке сброса увеличивалась;
2. При нажатии на кнопку «-» заданная скорость – уменьшалась;
3. При нажатии на кнопку сброса заданная скорость – обнулялась.

- При этом использовать все методы соединения сигналов и слотов.

Сигналы и слоты. Как соединить сигнал и слот?

Соединение объектов осуществляется при помощи статического метода connect():

```
QObject::connect (const QObject* sender,
                  const char*         signal,
                  const QObject*  receiver,
                  const char*         slot,
                  Qt::ConnectionType type = Qt::AutoConnection
                  );
```

`sender` – указатель на объект, отправляющий сигнал;

`signal` – сигнал, с которым осуществляется соединение, причем имя сигнала заключается в специальный макрос `SIGNAL(method());`

`receiver` – указатель на объект, который имеет слот для обработки сигнала;

`slot` – слот, который вызывается при получении сигнала. Прототип слота заключается в специальный макрос `SLOT(method());`

`type` – управляет режимом обработки*.

Сигналы и слоты. Как соединить сигнал и слот?

Пример соединения сигнала и слота:

```
PultWidget::PultWidget(QWidget *parent) :
    QWidget(parent)
{
    setupUi(this);
    //проинициализируем значения
    marshValue = 0;
    deltaMarshValue =0.1;

    //[4] при нажатии на кнопку QPushButton, она генерит сигнал clicked()
   //соединим кнопку "+" со слотом addMarshValue():
    //указатель на объект, содержащий слот addMarshValue - это this
    connect(btnPlusMarsh, SIGNAL(clicked()),this,SLOT(addMarshValue()));
    //в принципе синтаксис метода connect, когда происходит соединение со слотом,
    //принадлежащим классу, в котором происходит connect можно упростить и записать так:
    //connect(btnPlusMarsh, SIGNAL(clicked()),SLOT(addMarshValue()));
}
```

Сигналы и слоты. Как соединить сигнал и слот?

Альтернативный метод connect():

```
QObject::connect (const QObject*                sender,
                  const MetaMethod&             signal,
                  const QObject*                receiver,
                  const MetaMethod&             slot,
                  Qt::ConnectionType type = Qt::AutoConnection
                  );
```

Отличие от предыдущего метода в том, что сигнал и метод передаются через указатели на методы сигналов и слотов классов напрямую и без использования макросов `SIGNAL(method())` и `SLOT(method())`.

Для примера создадим слот под «-» значений

```
24      ····
25      ····//[5]·Создадим·слот·для·уменьшения·заданных·значений:
26      ····void·decMarshValue();
27      ····
28      ····
```

```
28
29      void·PultWidget::decMarshValue()
30      {
31      ····marshValue-=deltaMarshValue;
32      ····btnResetMarsh->setText(QString::number(marshValue));
33      }
34
```

Пример

Пример соединения сигнала и слота альтернативным способом:

```
18      ····
19      ····//[6].·соединим·сигнал·и·слот·по·второй·схеме
20      ····connect(btnMinusMarsh,&QPushButton::clicked,this,·&PultWidget::decMarshValue);
```

Особенность метода в том, что если вы ошибетесь с названием слотов и сигналов, ваша ошибка будет выявлена сразу в процессе компиляции программы.
Недостаток – то, что необходимо явно указывать имена классов для сигнала и слота и следить за совпадением их параметров.

Сигналы и слоты. Соединение с лямбдами.

1. [*capture*] (*params*) mutable exception attribute -> ret { body }
2. [*capture*] (*params*) -> ret { body }
3. [***capture***] (***params***) { body }
4. [***capture***] { body }

capture - определяет, какие символы, видимые в области объявления функции, будут видны внутри тела функции.

params - Список параметров, как в объявлении функции

ret - Возвращаемый тип. Если нет, то он выводится из возвращаемого значения (или void, если функция не возвращает никакого значения)

Сигналы и слоты. Соединение с лямбдами.

[*capture*] (*params*) { body }

capture - определяет, какие символы, видимые в области объявления функции, будут видны внутри тела функции.
Список символов может быть передан следующим образом:
[a,&b] где *a* захвачена по значению, а *b* захвачена по ссылке.
[this] захватывает указатель **this** по значению.
[&] захват всех символов по ссылке
[=] захват всех символов по значению
[] ничего не захватывает
params - Список параметров, как в объявлении функции
ret - Возвращаемый тип. Если нет, то он выводится из возвращаемого значения (или void, если функция не возвращает никакого значения)

Пример

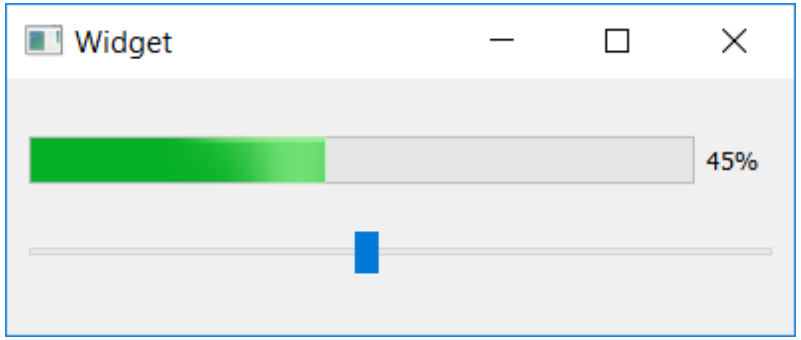
```
    //[7]- соединим кнопку Reset с использованием лямбда - функции
    connect(btnResetMarsh, &QPushButton::clicked, [this](){
        marshValue = 0;
        btnResetMarsh->setText("0");
    });
```

Практическая часть 2.

Для тех, кто подготовил панель управления ПА или первые прототипы своих пультов:

- соединить задатчики движения ПА с виджетами, выводящими заданные значения каждым из рассмотренных способов соединения сигналов и слотов.

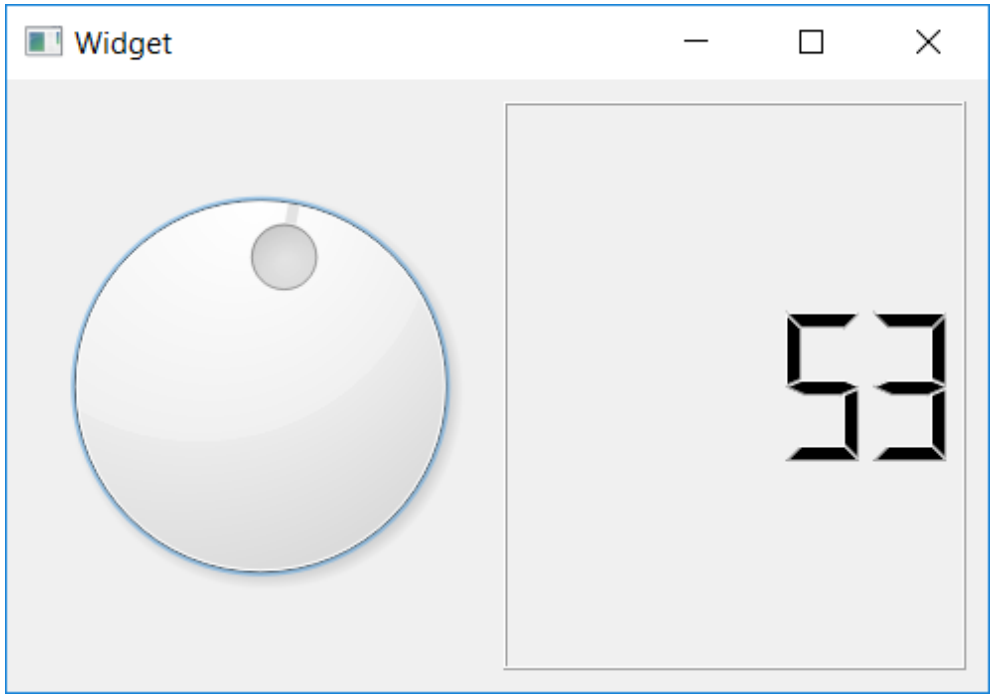
Задание 1



1. При изменении положения слайдера, должно соответственно измениться заполнение индикатора процесса
2. Подсказки:

- Сигнал изменения положения задатчика dial QSlider::valueChanged(int)
- Слот, который может изменить заполнение индикатора процесса – QProgressBar:: setValue(int)

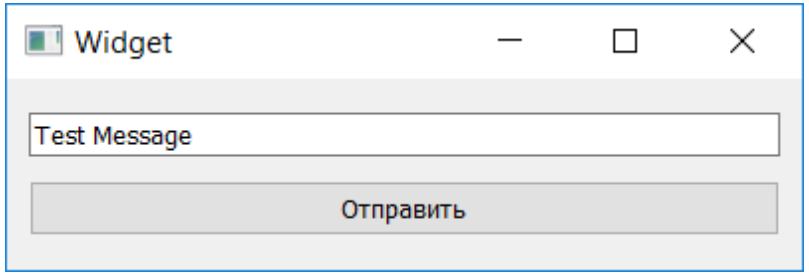
Задание 2

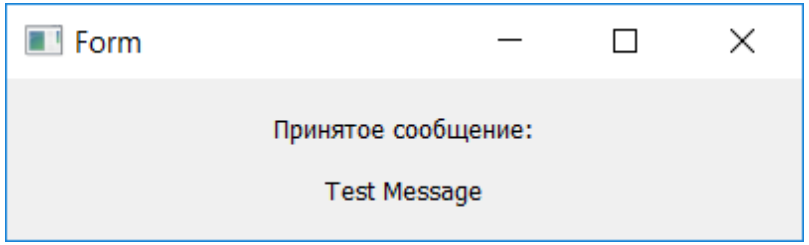


1. При изменении положения элемента dial, значение положения задающего элемента, должно отобразиться на цифровом индикаторе lcdNumber
2. Подсказки:

- Сигнал изменения положения задатчика dial QDial::valueChanged(int)
- Слот, который может вывести значение на цифровом индикаторе - QLCDNumber::display(int)- Вывести текст на метке можно с помощью

Задание 3





1. При нажатии кнопки «Отправить» сообщение, содержащееся в строке ввода окна «Widget» должно быть отображено в окне Form.
2. Подсказки:

- Сигнал нажатия кнопки QPushButton::clicked()
- Получить текст из строки ввода можно с помощью метода QLineEdit::text()
- Вывести текст на метке можно с помощью метода Qlabel::setText(QString)

Продвинутое использование сигналов и слотов

QSignalMapper – класс, который позволяет соединять один слот со многими, но различимыми сигналами.

Класс позволяет:

- собрать сигналы, не передающие данных;
- переслать их слоту, при этом передав идентификатор источника сигнала:
  - int id;
  - Qstring текст;
  - Указатель на объект Qwidget, QObject.

Продвинутое использование сигналов и слотов

- Установка соответствия сигнал – идентификатор (mapping) осуществляется методом setMapping();
- При появлении сигнала одного из объектов, объект QSignalMapping генеренрит сигнал mapped(выбранный тип идентификатора)

Пример 1

```
public slots:
    void addMarshValue();

    //[5] Создадим слот для уменьшения заданных значений:
    void decMarshValue();

    //[9] создадим слот под обработку кнопок изменения дифферента
    void changePitchValue(QString source);


private:
    //[7] Добавим новые переменные
    //переменная, которая хранит заданное значение по маршу
    double marshValue, pitchValue;
    //переменная, которая хранит значение шага изменения марша
    double deltaMarshValue, deltaPitchValue;

    //[8] создадим указатель на объект QSignalMapper и не забудем
    //сделать  #include <QSignalMapper> в самом начале
    QSignalMapper *signalMapper = nullptr;

};
```

Пример 1

```
{
    setupUi(this);
    //проинициализируем значения
    //[10]
    marshValue =pitchValue= 0;
    deltaMarshValue = deltaPitchValue = 0.1;
    //создадим объект signalMapper (выделим под него память)
    signalMapper = new QSignalMapper(this);
    //[4] при нажатии на кнопку QPushButton, она генерит сигнал clicked()
  //соединим кнопку "+" со слотом addMarshValue():
    //указатель на объект, содержащий слот addMarshValue - это this
    connect(btnPlusMarsh, SIGNAL(clicked()),this,SLOT(addMarshValue()));
    //в принципе синтаксис метода connect, когда происходит соединение со слотом,
    //принадлежащим классу, в котором происходит connect можно упростить и записать так:
    //connect(btnPlusMarsh, SIGNAL(clicked()),SLOT(addMarshValue()));

    //[6]. соединим сигнал и слот по второй схеме
    connect(btnMinusMarsh,&QPushButton::clicked,this, &PultWidget::decMarshValue);

    //[7]- соединим кнопку Reset с использованием лямбда - функции
    connect(btnResetMarsh, &QPushButton::clicked, [this](){ ...
    });
    //[11]
    //устанавливаем соответствие сигналов и идентификаторов
    signalMapper->setMapping(btnPlusPitch,"+");
    signalMapper->setMapping(btnMinusPitch,"-");
    signalMapper->setMapping(btnResetPitch,"Reset");

    connect(signalMapper, SIGNAL(mapped(QString)), this, SLOT(changePitchValue(QString)));
    
    connect(btnPlusPitch, SIGNAL(clicked()), signalMapper, SLOT(map()));
    connect(btnMinusPitch, SIGNAL(clicked()), signalMapper, SLOT(map()));
    connect(btnResetPitch, SIGNAL(clicked()), signalMapper, SLOT(map()));
}
```

```
void PultWidget::changePitchValue(QString source)
{
    if (source == "+") pitchValue+=deltaPitchValue;
    else if (source =="-") pitchValue-=deltaPitchValue;
    else if (source =="Reset") pitchValue = 0;
    btnResetPitch->setText(QString::number(pitchValue));
}
```

Пример 2

```
    //[13] изменяем тип connect : вместо:
    //connect(signalMapper, SIGNAL(mapped(QString)), this, SLOT(changePitchValue(QString)));
    //используем соединение через указатели:
    connect(signalMapper, (static_cast<void(QSignalMapper::*)(const QString &)>(&QSignalMapper::mapped)), \
    this, &PultWidget::changePitchValue);
```