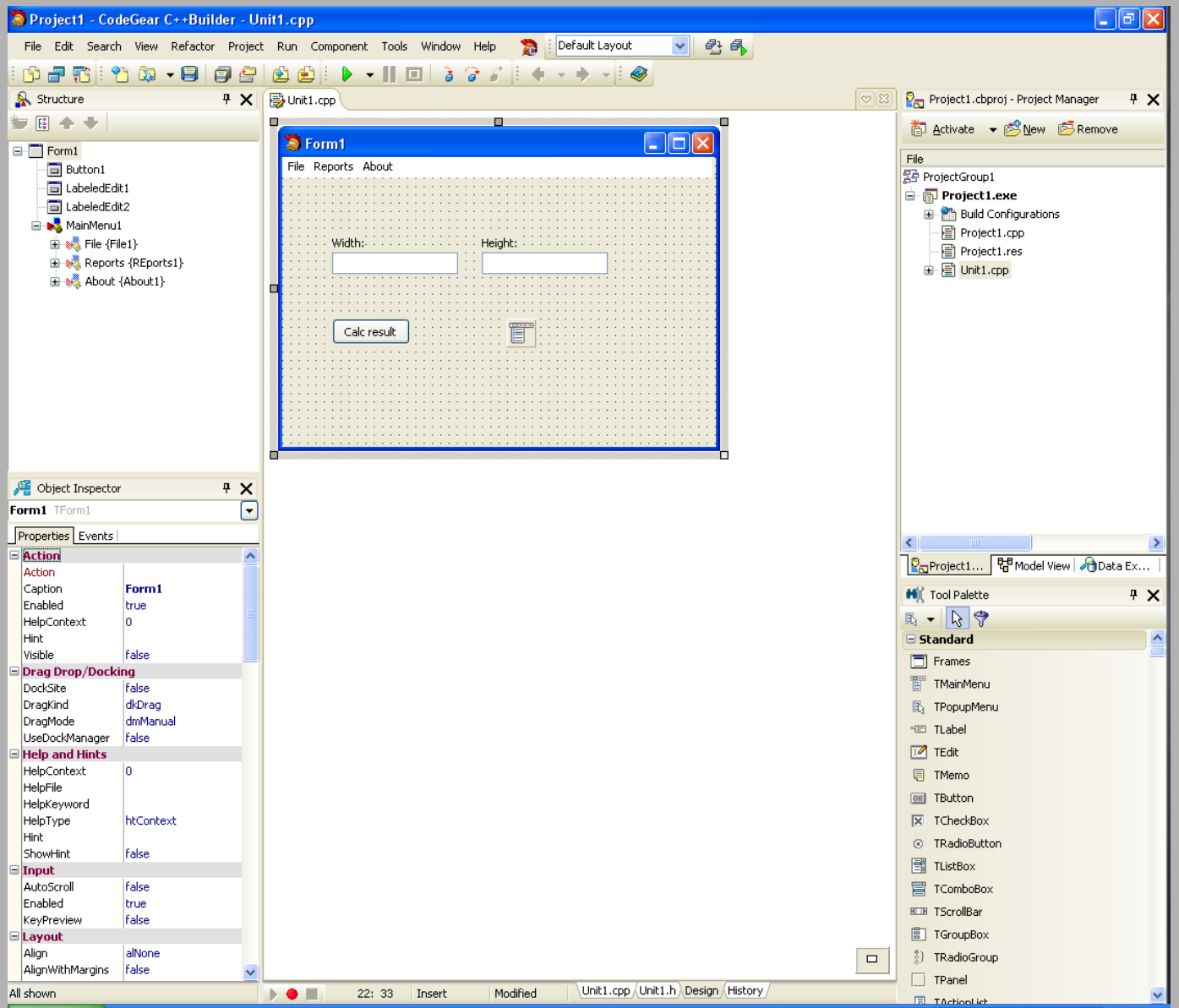
ПМО ИУС

RAD
(Rapid application Development)

Лекция 10

Rapid Application Development

Интегрированная среда разработки IDE – Integrated Development Environment



- Средства языка C++ интегрированы в визуальную среду разработки.

- Испытание прототипа приложения позволяет быстро переходить от прототипа приложения к полнофункциональному программному продукту. Это объединение трёх этапов

“редактирование => компиляция => сборка”

в единый производственный процесс.

- Drag-and-drop - выполнение ряда функций посредством простого перетаскивания захваченных мышью компонентов и элементов среды RAD.

- Two-way-tools - обеспечивает контроль за кодом с помощью интегрированного и гибкого взаимодействия между инструментами визуального проектирования и редактором кода.

- Библиотека визуальных компонент (VCL – Visual Component Library) – содержит более 130 повторно используемых компонент, которые реализуют все элементы пользовательского интерфейса Windows, а также представляют в распоряжение компоненты для работы с базами данных, со средствами мультимедиа, системные инструменты, сетевые решения, диалоговые средства и многое другое.

VCL – Visual Component Library

- Кроме того при программировании в C++ Builder компоненты можно использовать как непосредственно, так и менять их свойства, облик и поведение, порождать от них производные элементы, обладающие нужными характеристиками.

Свойства, методы и события

- Свойства (Properties) – характеристики используемых объектов (поля класса / объекта).
- Методы (Methods) – производят некоторые операции над объектом (методы класса / объекта).
- События (Events) – связывают воздействия пользователя на объекты с кодами реакции на эти воздействия.

Свойства, методы и события

- События могут возникать при нажатии клавиш, перемещении мыши, по прерываниям таймера, при изменении состояний объектов. Работая повсеместно, свойства, методы и события образуют среду RAD с точки зрения ООП.

Свойства

- Свойства являются атрибутами компонента, определяющими его внешний вид и поведение. Они присущи только компонентам и представляют собой расширение понятия членов данных.
- С точки зрения программиста свойства выглядят как обычные переменные: можно считывать или присваивать им значения как с помощью инспектора объектов, так и во время выполнения программы.

Свойства

- Операции со свойствами по своему поведению перегружены, т.е. изменение свойств может вызвать достаточно сложные операции в среде приложения, но в большинстве случаев пользователь этого не замечает.
- Свойства пользователи могут задавать на стадии проектирования до непосредственного запуска приложения.

Свойства

- Свойство может прятать детали реализации от пользователя и иметь ‘нормальный’ вид только при установке и записи значений.
- Свойство в процессе работы имеет 4 атрибута:

read – метод чтения или член данных.

write – метод записи или член данных.

default – значение по умолчанию.

stored – текущее значение.

События

- представляют собой средства, с помощью которых компонент сообщает пользователю о том, что на него оказано некоторое предопределённое воздействие.
- Работа ОС Windows предполагает событийную модель работы приложения, т.е. в приложении описываются реакции на те или иные события, а сама ОС всё время занимается постоянным опросом с целью выявления наступления какого-либо события.

События

- представляют собой механизм связи происшествия с некоторым кодом.
- С точки зрения программиста событие представляет собой имя, которому можно присвоить некоторый вызываемый метод и программист воспринимает событие как способ указания к какому из написанных методов должна обращаться программа в случае, когда данное событие происходит.

События

- ОС имеет всегда список возможных событий и если ни одно из них не наступило, находится в режиме ожидания. Если некоторое событие произошло, то ОС передаёт управление обработчику события до тех пор, пока обработка не завершится. Только после завершения обработки произойдёт обработка следующего.

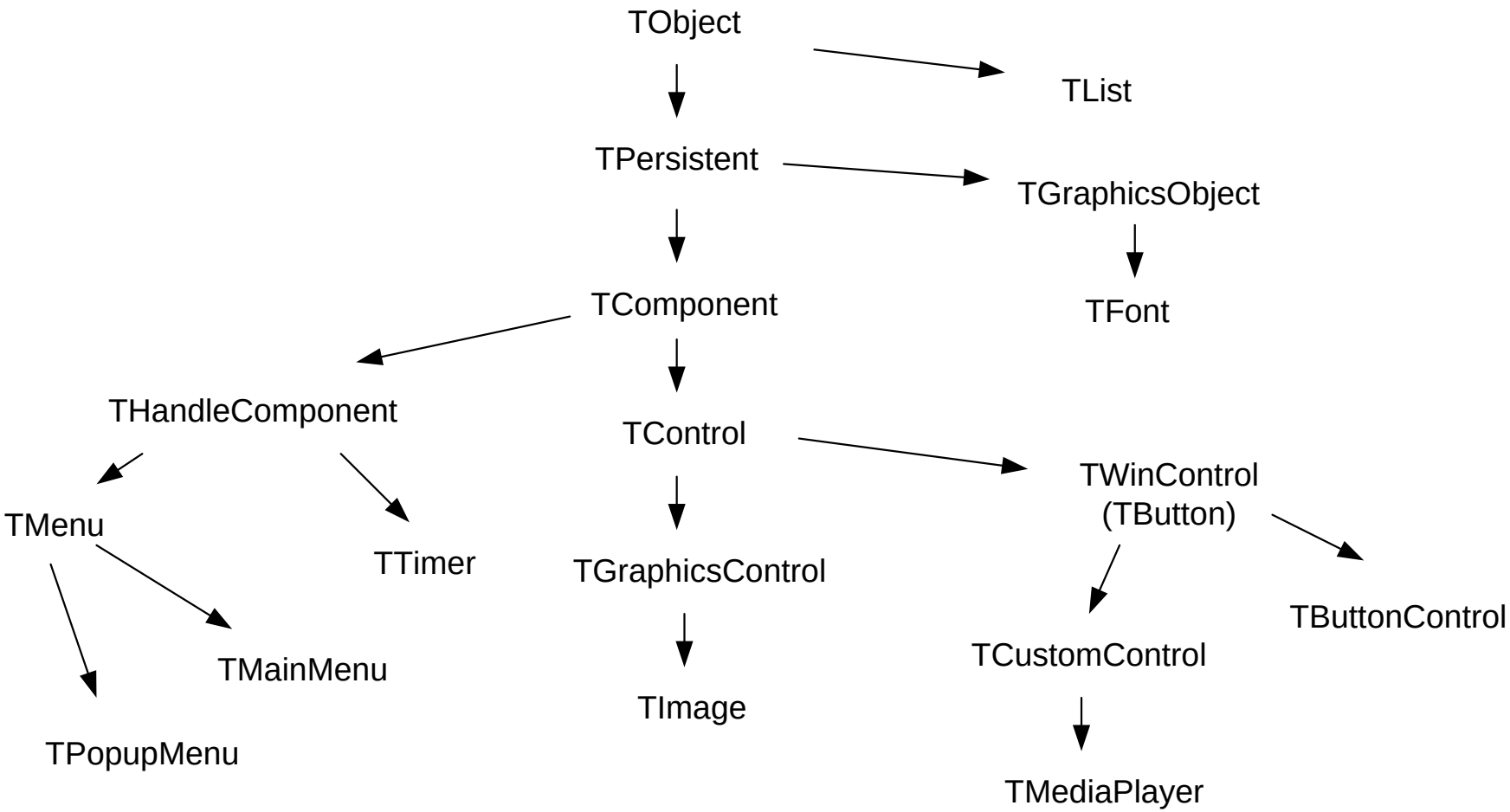
VCL – Visual Component Library

- Компоненты VCL – это строительные кирпичи, из которых конструируется приложение в среде RAD.
- Все компоненты разделяются на 4 типа:
- 1) Стандартные – имеют поведение типовых элементов ОС Windows (Standard и Win32 вкладки).
- 2) Оригинальные – элементы управления, у которых нет методов собственного отображения, ни заранее определённого поведения. (TPanel)
- Стандартные и оригинальные являются оконными компонентами. Они могут быть активизированы, используют системные ресурсы, и могут быть контейнерами.

VCL – Visual Component Library

- 3) Графические – видимые элементы управления, которые не могут принять фокус ввода. Обеспечивают отображение объектов без использования системных ресурсов (TImage, TShape) и не могут служить контейнерами для других компонентов.
- 4) Невидимые – не появляются на форме в виде каких-либо элементов управления, их поведение определяется на этапе проектирования и может быть активизировано во время выполнения программы. (TTable, TOpenDialog)

Иерархия классов VCL



Состав проекта

cbproj	конфигурация проекта
cpp	исполняемый код реализации
h	интерфейсы компонент
dfm	ресурсы ресурсы форм
res	ресурсы всего приложения
Debug/*.*	файлы компиляции и выполнения отладки
__history/*.*	файлы отката

Класс AnsiString (String)

<, >, =, ==, !=, >=, <=, +, +=

c_str()	указатель char* на символьный С-массив
Length()	длина строки в байтах
IsEmpty()	true если строка пуста
Insert	вставка текста в указанную позицию
Delete	удаление указанных символов
Pos	поиск подстроки
LowerCase	возврат строки в нижнем регистре
UpperCase	возврат строки в верхнем регистре
Trim	очистка ведущих и хвостовых управляющих символов и пробелов
ToInt	преобразование строки к int
ToDouble	преобразование к double (float)
Substring	получение подстроки

Функции-помощники: **StrToInt, IntToStr, StrToFloat, FloatToStr**.

Типы свойств

Simple	простые числовые, символьные и строчные свойства
Enumerated	свойства перечисляемого типа – выбор из списка перечисляемых значений
Set	множество, внутри которого каждый элемент может быть как **true**, так и **false**
Object	свойства, которые сами по себе являются объектами
Array	обслуживаются своими собственными редакторами свойств

Свойства

Name имя компонента

Tag пользовательское число

Visible видимость компонента

Enabled возможность получения фокуса ввода и генерации событий от пользователя

Hint комментирующая подсказка (содержание)

ShowHint включение/выключение

Left, Width, Top, Height

положение компонента относительно родителя (**Parent**)

Align выравнивание

Autosize автоматическое изменение размера

Свойства

Caption	заголовок компонента
Text	текстовое содержимое
TabOrder	порядок смены фокуса по TAB
TabStop	остановка при TAB-смене фокуса
Font	свойства шрифта компонента
Charset	кодировка
Color	цвет символов
Height	высота в пикселах
Name	имя шрифта
Style	стиль (**Bold**, *Italic*, Uderline, Strike)

События

OnClick активизация действия
OnDblClick двойной клик мыши

OnMouseDown нажатие кнопки мыши
OnMouseUp отпуск кнопки мыши
OnMouseMove движение мыши над компонентом
OnMouseWheel реакция на колесо мыши
OnMouseWheelUp движение колеса мыши вверх
OnMouseWheelDown движение колеса мыши вниз

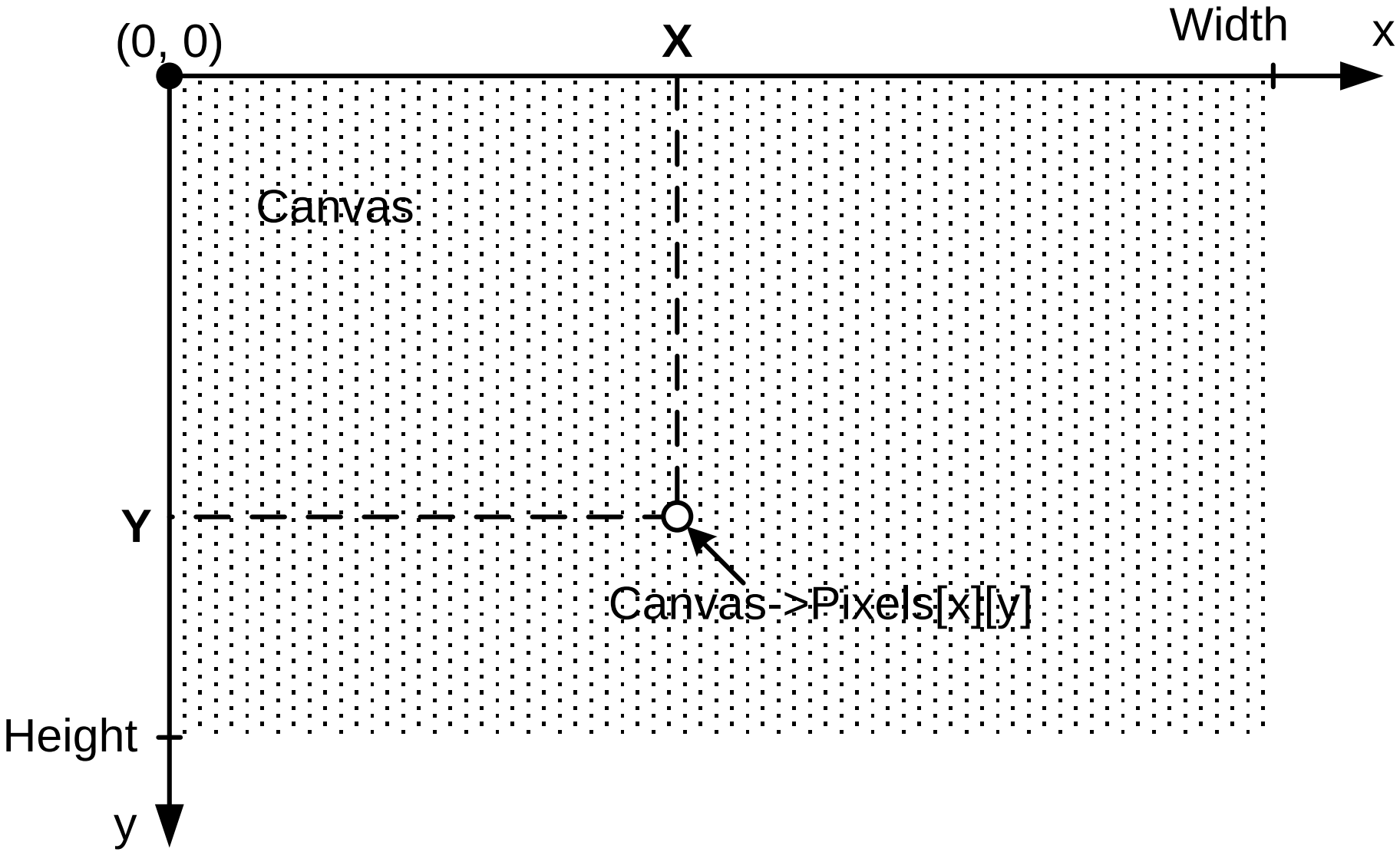
OnDragOver перетаскивание элемента
OnDragDrop отпуск при перетаскивании элемента

OnKeyDown нажатие клавиши на клавиатуре
OnKeyUp отпуск клавиши на клавиатуре
OnKeyPress нажатие алфавитно-цифровой клавиши

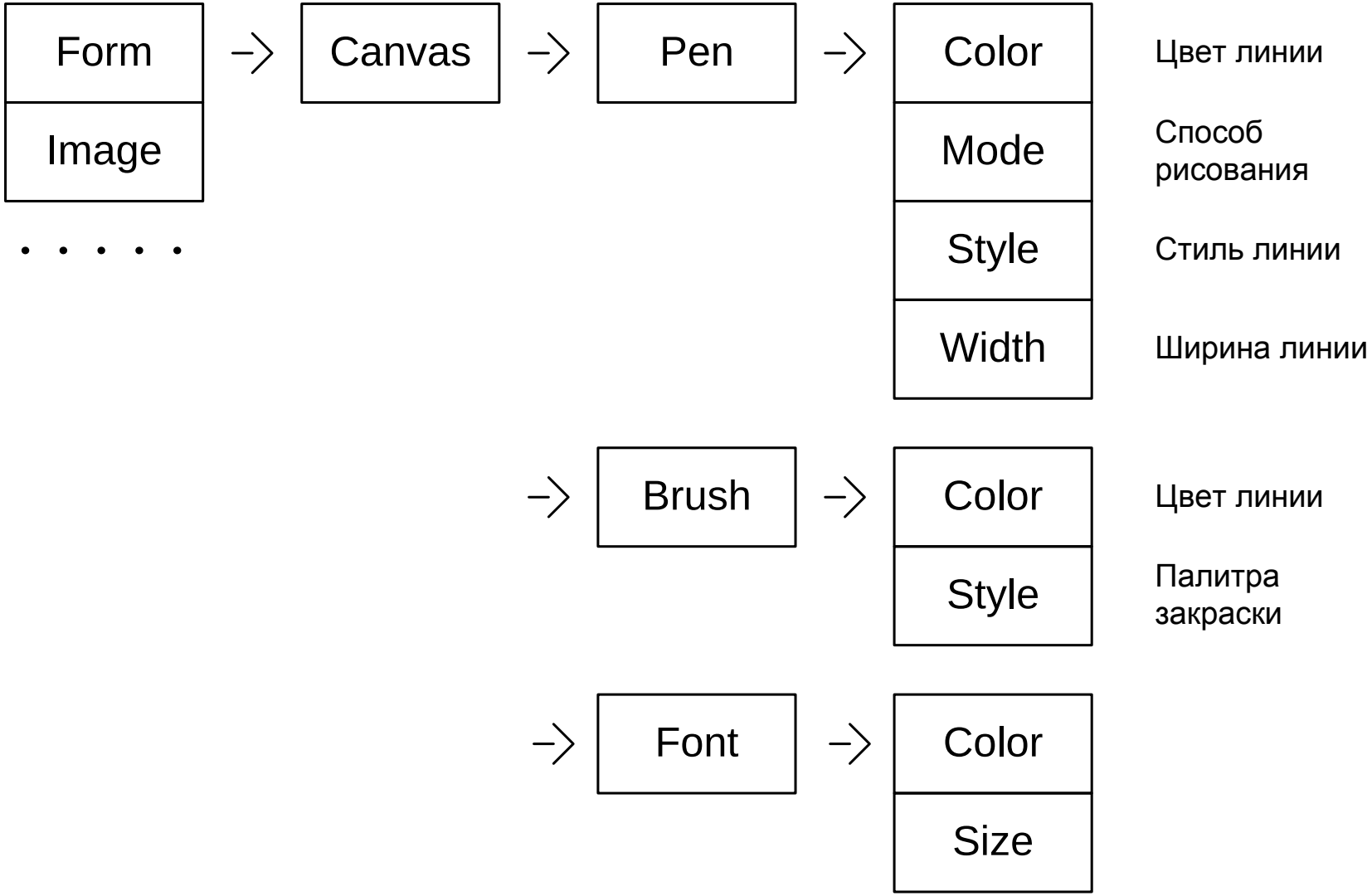
События

OnTimer	событие таймера
OnChange	изменение значения
OnExit	потеря фокуса ввода
OnEnter	получение фокуса ввода
OnClose	закрытие формы
OnCloseQuery	попытка закрытия формы
OnCreate	создание формы
OnShow	событие показа
OnPaint	событие перерисовки
OnResize	изменение размеров
OnHide	скрытие (например, в трей)

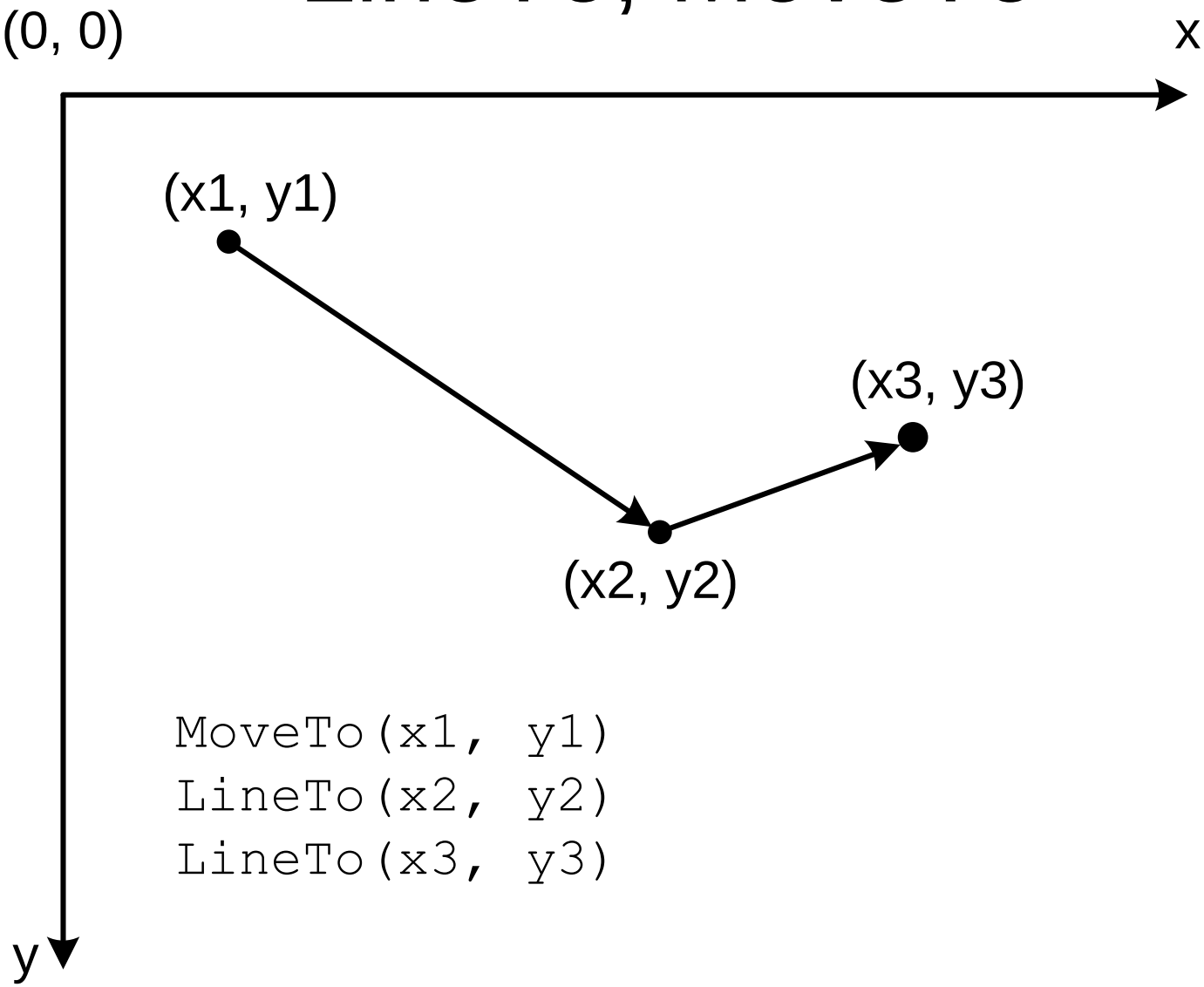
Canvas



Canvas

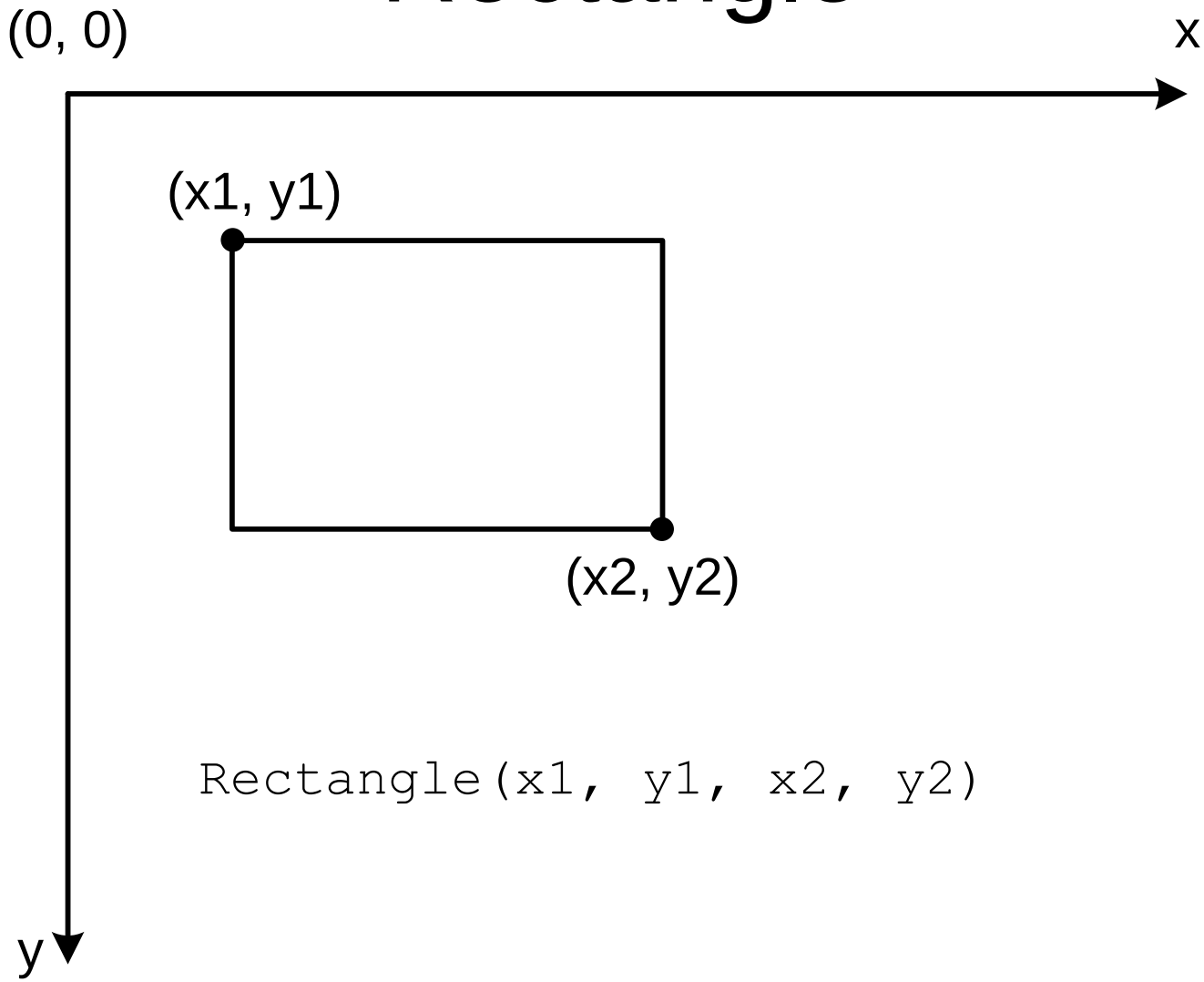

Form
Image
. . . . .
Canvas
Pen
Color
Цвет линии
Mode
Способ рисования
Style
Стиль линии
Width
Ширина линии
Brush
Color
Цвет линии
Style
Палитра закраски
Font
Color
Size

LineTo, MoveTo



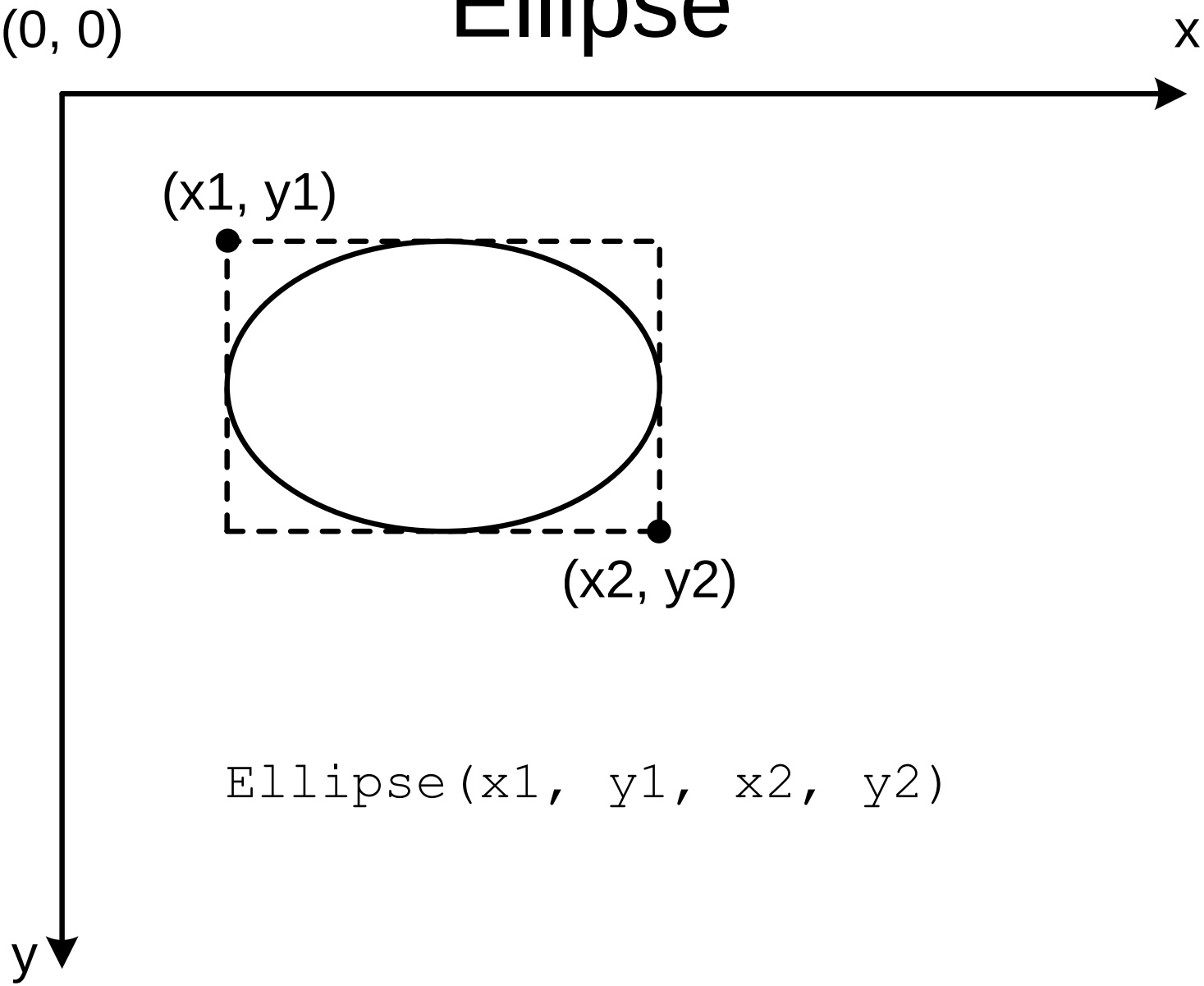

```
MoveTo(x1, y1)
LineTo(x2, y2)
LineTo(x3, y3)
```

Rectangle



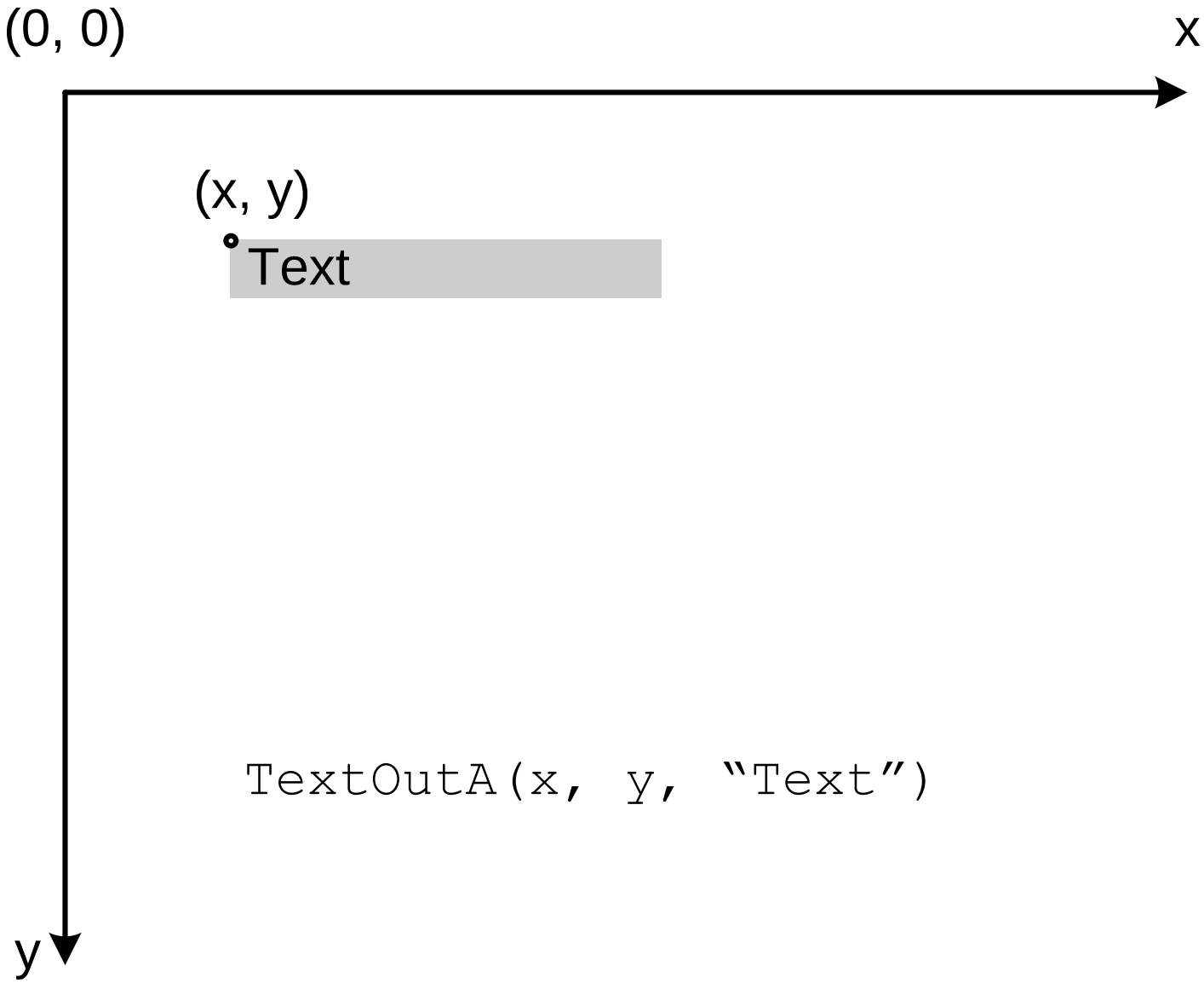

```
Rectangle(x1, y1, x2, y2)
```

Ellipse




```
Ellipse(x1, y1, x2, y2)
```

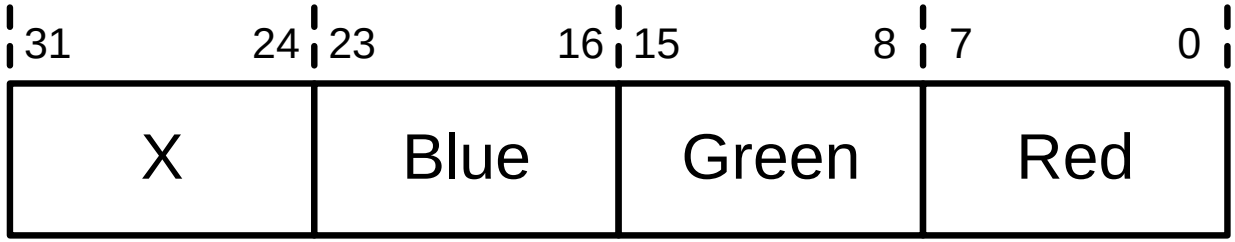
TextOutA




```
TextOutA(x, y, "Text")
```

TColor

long int – 32 bit




```
0x0080a0ff :
Blue  80 : 128
Green a0 : 160
Red   ff : 255
```

Стандартные цвета:

```
clBlack  : 0x0000000
clWhite  : 0x0ffffff
clRed    : 0x00000ff
clGreen  : 0x000ff00
clYellow : 0x000ffff
```

User Dialogs

```
void ShowMessage(constAnsiString Msg);
void ShowMessagePos(constWideString Msg, int X, int Y);
```

```
WideString InputBox(
   constWideString ACaption,
   const WideString APrompt,
   const WideString ADefault );

double InputBox(
   const WideString ACaption,
   const WideString APrompt,
   int ADefault,
   int Min = std::numeric_limits<int>::min(),
   int Max = std::numeric_limits<int>::min(),
   int Increment = 1 );
```

```
double InputBox(
        const WideString ACaption,
        const WideString APrompt,
        double ADefault,
        double Min =
std::numeric_limits<int>::min(),
        double Max =
std::numeric_limits<int>::max(),
        int Decimals = 1 );
```

ACaption текст заголовка.
APrompt текст запроса.
ADefault текст по умолчанию.

```
AnsiString InputString = InputBox("Input Box", "Prompt",
   "Default string");
```

User Dialogs

```
enum TMsgDlgType {
   mtWarning,
   mtError,
   mtInformation,
   mtConfirmation,
   mtCustom
};

enum TMsgDlgBtn {
   mbYes,
   mbNo,
   mbOK,
   mbCancel,
   mbAbort,
   mbRetry,
   mbIgnore,
   mbAll,
   mbNoToAll,
   mbYesToAll,
   mbHelp
};
```

```
CreateMessageDialog(
    <сообщение>,
    <тип диалога>,
    <множество кнопок>);
int
MessageDlg(
    const WideString Caption,
    const WideString Msg,
    TMsgDlgType DlgType,
    TMsgDlgButtons Buttons,
    int HelpCtx,
    int X,
    int Y,
    TMsgDlgBtn DefaultBtn = mbNone,
    TBitmap Bitmap = (Qgraphics::TBitmap*) NULL);
int
MessageDlg(
    "The list has " + IntToStr(pList->Count) + " objects",
    mtInformation,
    TMsgDlgButtons() << mbOK,
    0);
int
MessageDlgPos(
    "Are you there?",
    mtConfirmation,
    TMsgDlgButtons() << mbYesNoCancel,
    0,
    100,
    200);
```

User Dialogs

```
bool
PromptForFileName(
   AnsiString &AFileName,
   constAnsiString AFilter,
   const AnsiString ADefaultExt,
   const AnsiString ATitle,
   const AnsiString AInitialDir,
   bool SaveDialog );

 if (
        MessageDlg(
            "Are you sure?",
            mtConfirmation,
            TMsgDlgButtons() << mbYes << mbCancel << mbNo << mbHelp,
            0) == mrYes
        ) {
        Button1->Caption = "";
    }
```