**Занятие 1**

**Повторение. Общие сведения**

1. Что такое JVM?

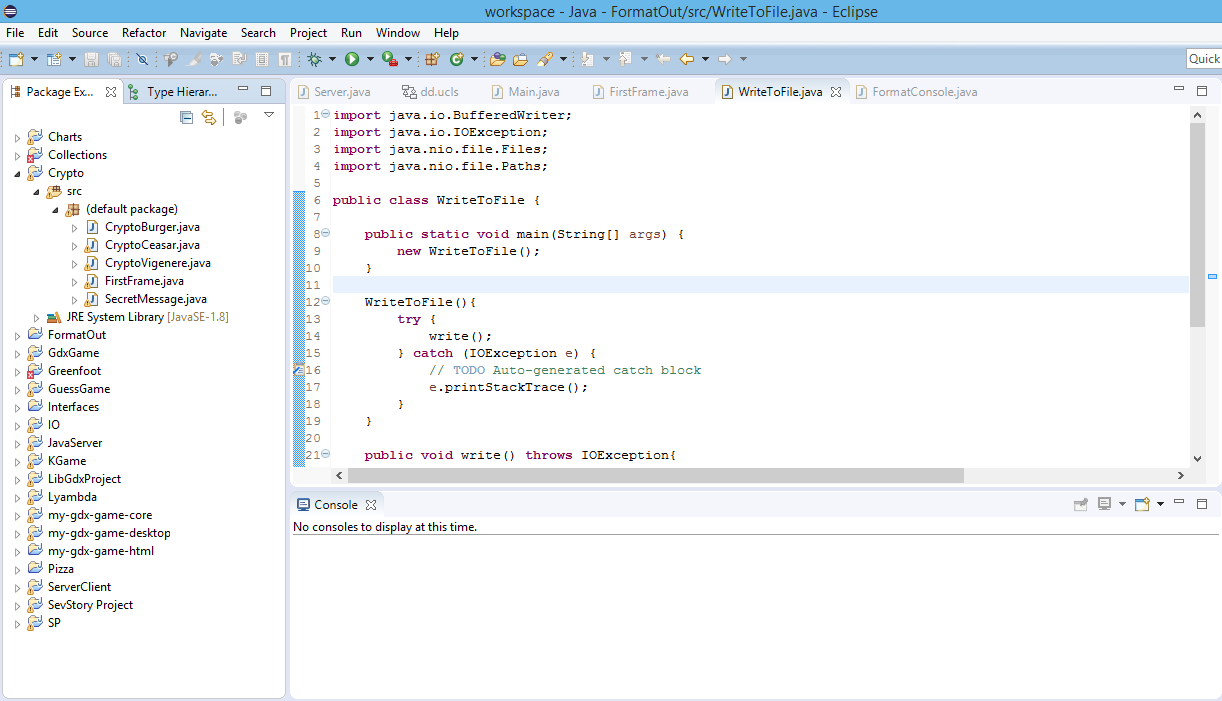
JVM – это Java Virtual Machine, виртуальная машина Java,  
интерпретирующая байт-код, описываемый в class-файлах. Ее применение  
необходимо для обеспечения кросс-платформенности, а также для  
безопасности.

1. Является ли язык Java компилируемым или интерпретируемым?  
   a.) Используются оба подхода. Исходный код сначала компилируется в байткод, который затем интерпретируется виртуальной машиной.
2. 1-3. Что такое механизм автоматической сборки мусора (garbage collector)?  
   a.) Этот механизм автоматически подсчитывает количество ссылок на каждый  
   объект Java. Когда на объект больше не указывает ни одна ссылка, он удаляется из памяти, освобождая ресурсы для программы.
3. Из чего состоит и в каком виде записывается программа, написанная на Java?  
     
   Программа на языке Java состоит из классов, записанных в  
   текстовых файлах с расширением java. Затем с помощью компилятора  
   генерируются бинарные .class-файлы, содержащие байт-код, который  
   затем интерпретируется виртуальной машиной.
4. Что такое JDK и JRE? В чем сходство и разница между ними? Какие основные  
   утилиты входят в их состав?  
   a.) JDK – это Java Development Kit, средство разработчика Java, включающее  
   в себя набор утилит, стандартные библиотеки с их сходным кодом и набор  
   демонстрационных примеров. Утилиты включают в себя:  
   • java – реализация JVM  
   • javac – компилятор Java  
   • jar – архиватор формата JAR  
   • javadoc – утилита для автоматической генерации документации  
   JRE – это Java Runtime Environment, среда выполнения Java,  
   предназначена только для запуска готовых Java-приложений, а потому  
   содержит лишь реализацию виртуальной машины и набор стандартных библиотек.

Работа в среде программирования Eclipse

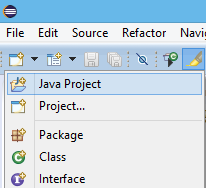
Текущая версия Eclipse Neon, но уже скоро ежегодное обновление.

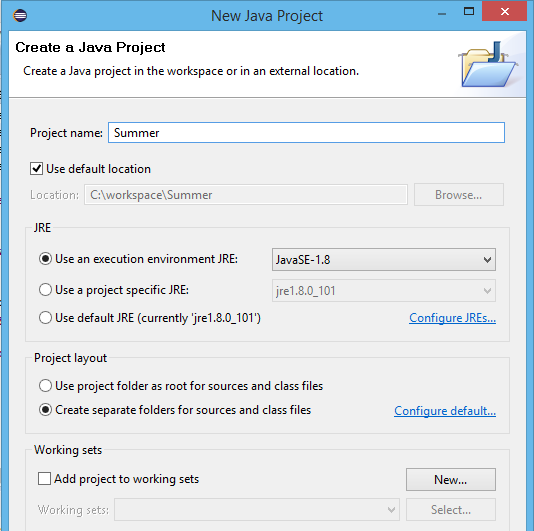
<https://www.eclipse.org/downloads/>



Принципиальное отличие от DrJava:

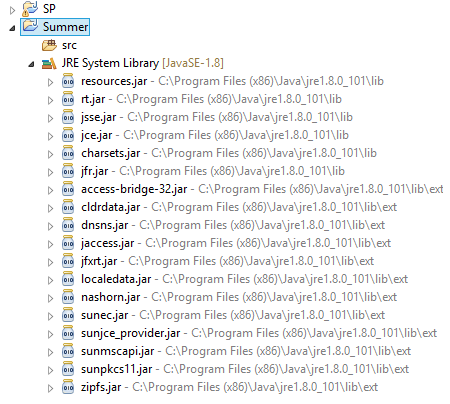
1. Создается проект
2. Создаются классы проекта





Файлы проекта

Посмотрим, что получилось:

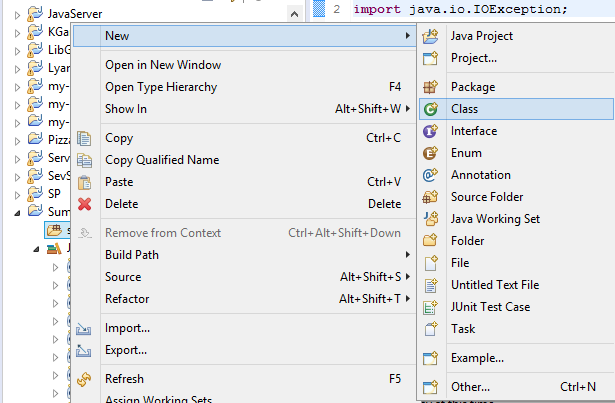


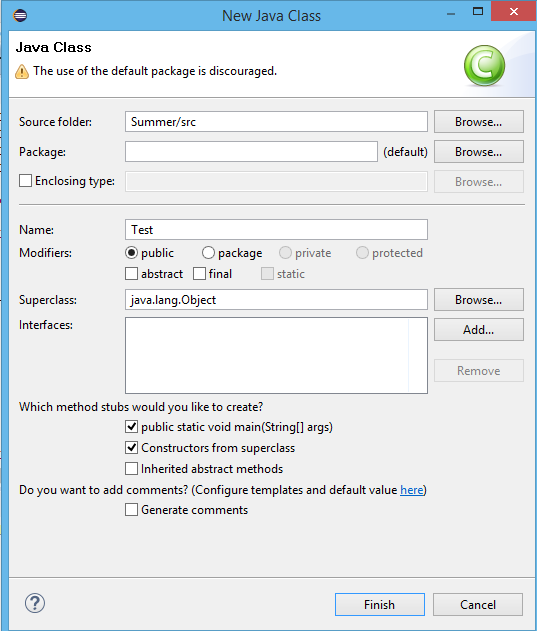
Здесь будут классы и все остальное

Это все родное и нужное.

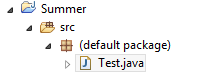
Можно добавлять всякие-разные библиотеки

Создадим новый класс и начнем работать.





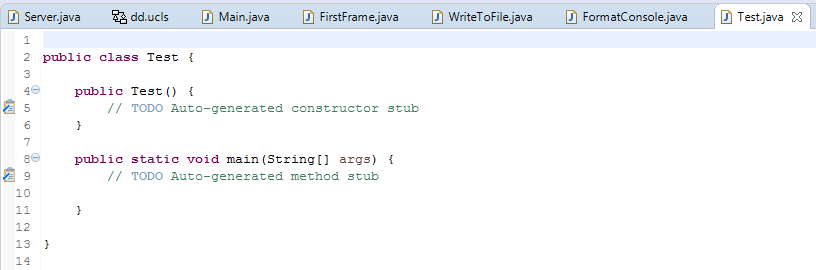
Что это?



Вот это!

**Java package** (пакет **Java**) — механизм, позволяющий организовать **Java** классы в пространстве имен аналогично модулям в языке программирования Модула. **Java** пакеты могут содержаться в сжатом виде в JAR файлах. Обычно в пакеты объединяют классы одной и той же категории, либо предоставляющие сходную функциональность.

Ладно, поехали дальше.



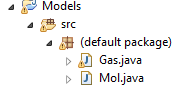
Практическая работа

Вывести в консоль строку «Привет из эклипса!» или что угодно ☺

**Практическая работа**

**Модель идеального газа**

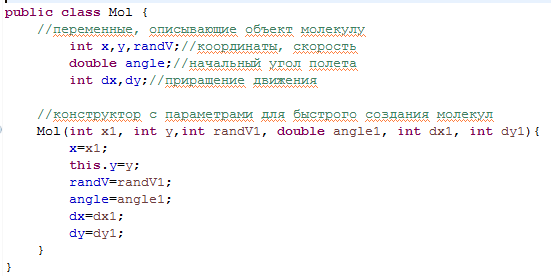
В физике идеальным называется газ, в котором силы взаимодействия между частицами пренебрежимо малы.



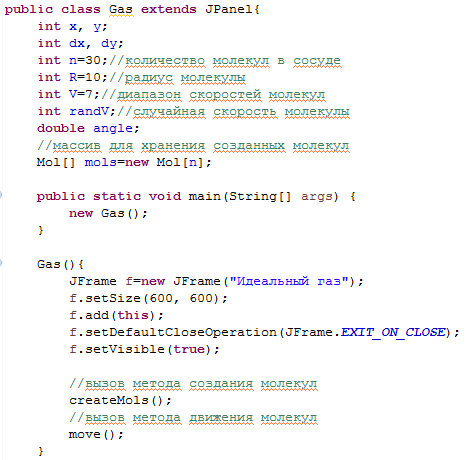
Проект состоит из двух классов.

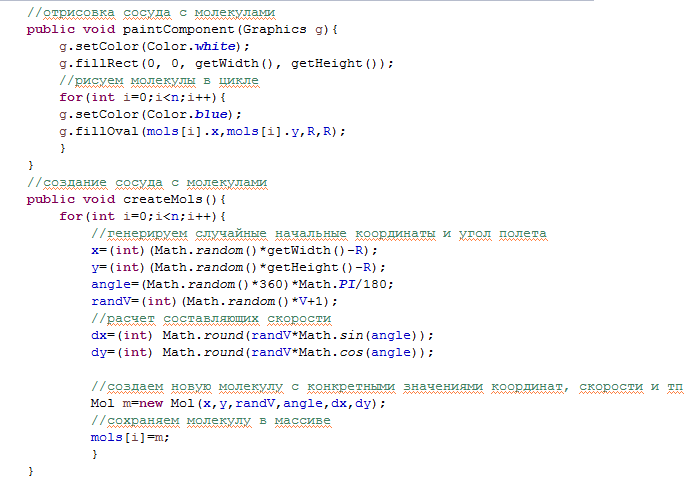
Класс Mol – описывает объект-молекулу.

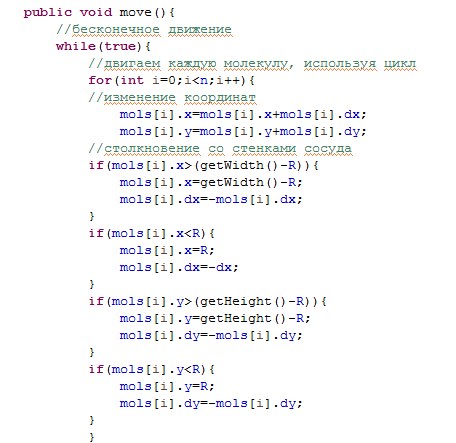
Класс Gas – класс с методом main(). В нем создается определенное количество двужущихся молекул.

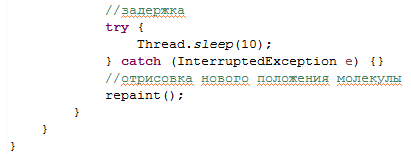


Подумаем хорошенько…









**Занятие 2**

**Работа с WindowBuilder в Eclipse**

**WindowBuilder** - is a powerful and easy to use bi-directional Java GUI designer.

WindowBuilder is composed of SWT Designer and Swing Designer and makes it very easy to create Java GUI applications without spending a lot of time writing code. Use the WYSIWYG (**W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et) visual designer and layout tools to create simple forms to complex windows; the Java code will be generated for you. Easily add controls using drag-and-drop, add event handlers to your controls, change various properties of controls using a property editor, internationalize your app and much more.

WindowBuilder is built as a plug-in to Eclipse and the various Eclipse-based IDEs (RAD, RSA, MyEclipse, JBuilder, etc.). The plug-in builds an abstract syntax tree (AST) to navigate the source code and uses GEF (Graphical Editing Framework) to display and manage the visual presentation.

Generated code doesn't require any additional custom libraries to compile.

and run: all of the generated code can be used without having WindowBuilder Pro installed. WindowBuilder Pro can read and write almost any format and reverse-engineer most hand-written Java GUI code. It also supports free-form code editing (make changes anywhere...not just in special areas) and most user re-factorings (you can move, rename and subdivide methods without a problem).

Standard Widget Toolkit, или **SWT**— библиотека с открытым исходным кодом для разработки графических интерфейсов пользователя на языке Java. Разработана фондом Eclipse, лицензируется под Eclipse Public License, одной из лицензий открытого ПО.

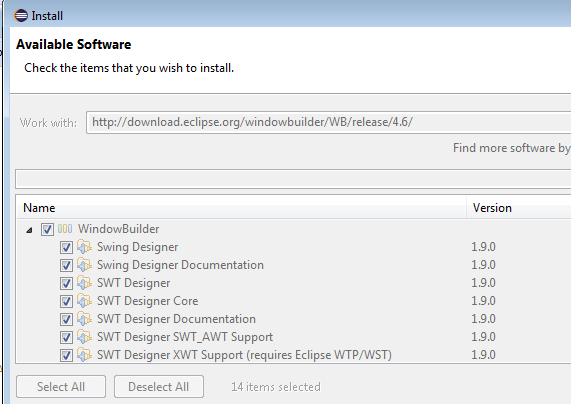
SWT не является самостоятельной графической библиотекой, а представляет собой кросс-платформенную оболочку для графических библиотек конкретных платформ, например, под Linux SWT использует библиотеку GTK+. SWT написана на стандартной Java и получает доступ к OS-специфичным библиотекам через Java Native Interface, который рассматривается в качестве сильного средства, несмотря на то, что это не является чистой Java.

SWT — альтернатива AWT и Swing (Sun Microsystems) для разработчиков, желающих получить привычный внешний вид программы в данной операционной системе. Использование SWT делает Java-приложение более эффективным, но снижает независимость от операционной системы и оборудования, требует ручного освобождения ресурсов и в некоторой степени нарушает Sun-концепцию платформы Java.

Установка:

Help – Install New Software, ввести ссылку и нажать Enter:

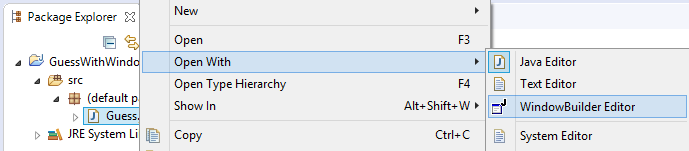
<http://download.eclipse.org/windowbuilder/WB/integration/4.6/>



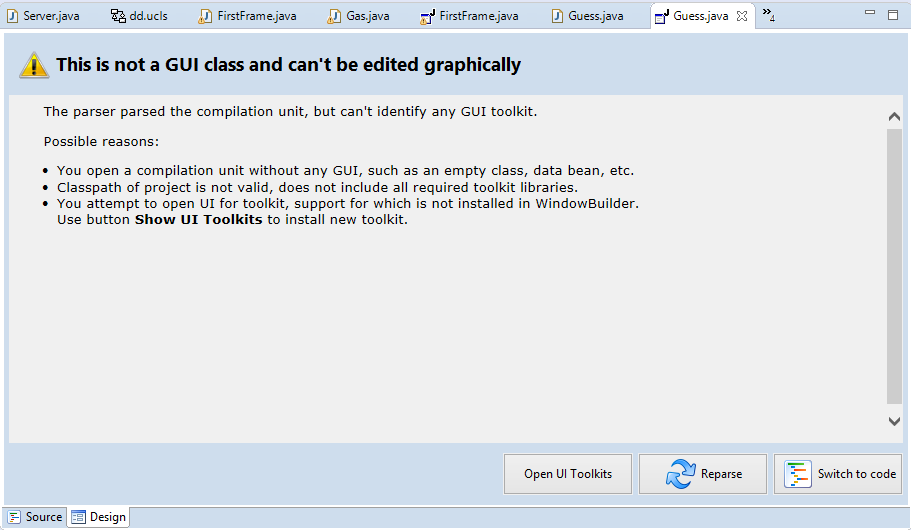
Теперь вы можете открыть любой свой класс в этом визуальном редакторе.

Допустим, я создаю класс Guess.

Открываю класс с помощью WindowBuilder:



Что же это такое? ☹

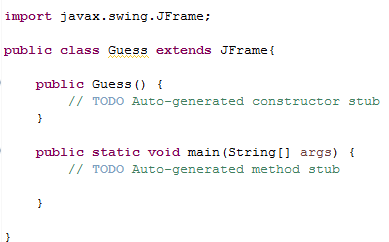


Переключимся на вкладку Source:

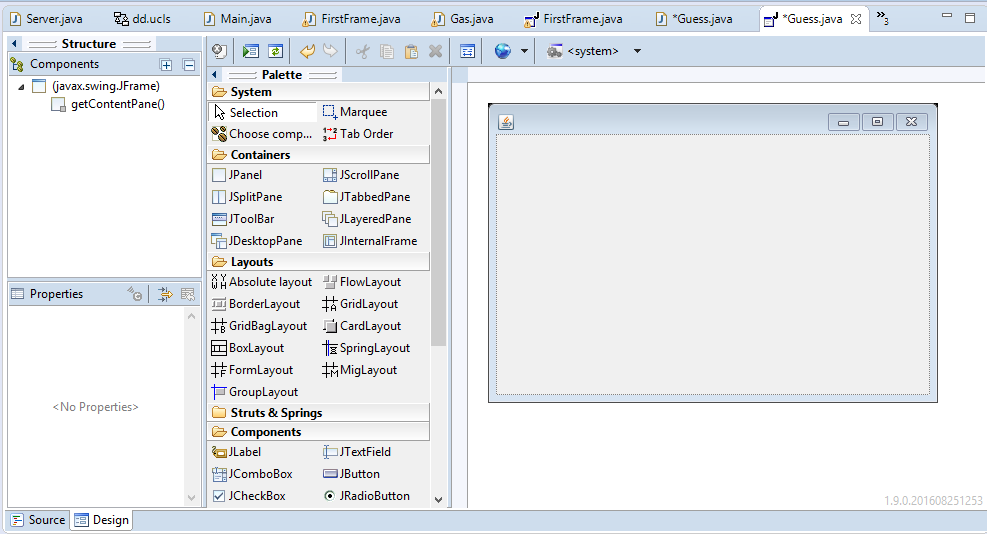




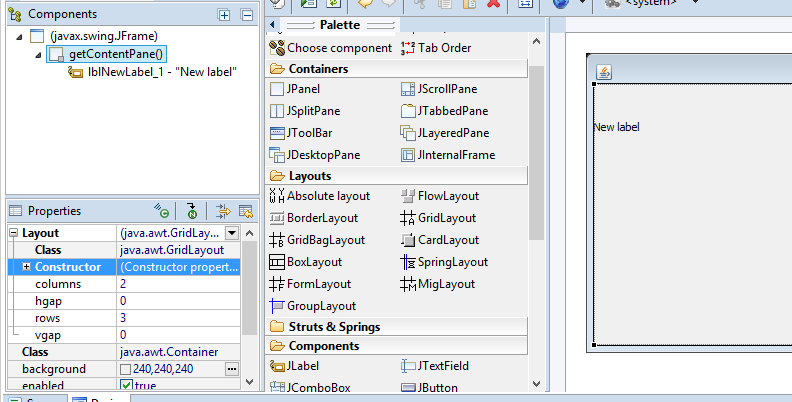
И превратим наш класс в GUI-класс:



Переключимся обратно. Вид дизайнера:



Связь кода и графического отображения:



**public** **class** Guess **extends** JFrame{

**public** Guess() {

getContentPane().setLayout(**new** GridLayout(3, 2, 0, 0));

JLabel lblNewLabel\_1 = **new** JLabel("New label");

getContentPane().add(lblNewLabel\_1);

}

Недостатки:

1. Код всех элементов содержится в конструкторе, т.е. «все в кучу».
2. Имена элементов генерируются автоматически, могут не соответствовать логике программы.
3. Элементы управления объявлены в конструкторе, т.е. к ним нет доступа из других методов.

**Практическая работа**

1. Разработать дизайн оконного приложения для игры «Угадай число от 1 до 100» с подсказками больше-меньше.
2. Написать код для работы программы.

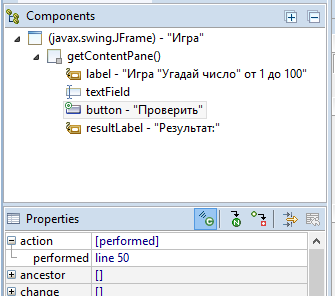
Примечание:

Приведите в порядок имена элементов управления, особенно тех, которые будут задействованы в коде: текстового поля, кнопки, метки.

Продумайте названия и типы переменных для хранения: загаданного рандомного числа, числа, введенного пользователем и тп.

Используйте методы getText() и setText(), чтобы считать и записать данные в элемент, и сохраните эти данные в соответствующих переменных.

Чтобы добавить анонимный класс-слушатель:

1. В окне редактора свойств переключиться на вид Show Event
2. 

**Занятие 3**

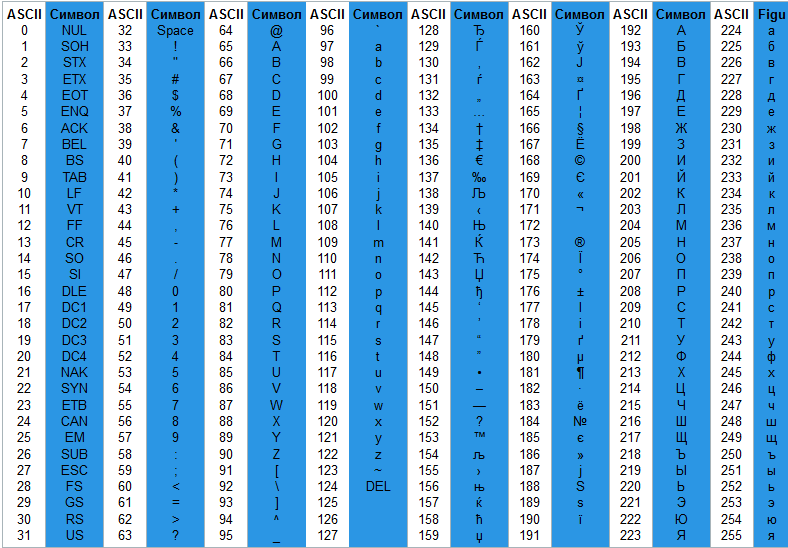
**Шифрование данных**

Криптография, наука о шифровке и дешифровке информации. На самом деле зашифрованные послания существуют с древних времён, и использовались армиями для передачи секретных сообщений.

Посмотрим на один из простейших шифров — шифр Цезаря, названный в честь римского императора. В этом шифре каждая буква текста заменяется на другую, которая находится на фиксированное число букв ниже в алфавите. Это фиксированное число (сдвиг) называется **ключом**. Так, ключ 1 переводит букву латиницы C в букву D, а Z — по циклу в A. Если ключ 3, то буква C перейдет в F, а Z — в C.   
  
**Примеры:** используем шифр Цезаря с ключом 5 на слове cat.

c > h  
a > f  
t > y   
Caesar (cat, 5) = hfy

Для программирования шифра используется таблица:



Помним, что строка — это просто массив символов char.

message.charAt(i)

Как получить номер символа в таблице ASCII:

(**int**)message.charAt(i)

Затем в строке

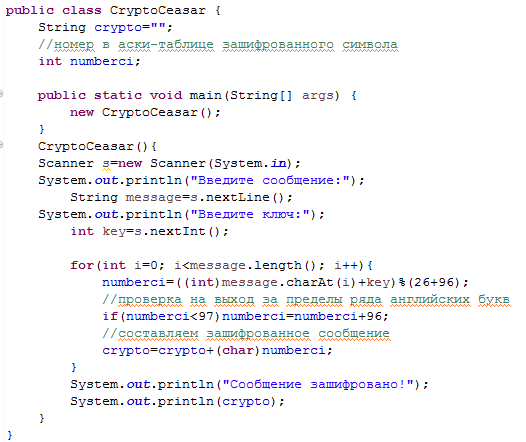
*int result = (letter + key);*

мы прибавляем значение ключа к ASCII-коду буквы и сохраняем их в переменной целого типа. Это будет ASCII-код зашифрованного символа.

Обязательно делаем проверку на выход за пределы алфавита (переходим в начало алфавита и отсчитываем от начала).

Шифр Цезаря прост, но ненадёжен. Для английского алфавита — всего 26 вариантов шифровки, перебрать все варианты легко даже без компьютера. Тем не менее, шифр Цезаря часто используют в качестве шага в других шифрах.

**Пример**



**Практическая работа**

1. Получить ключ
2. Получить текст
3. Зашифровать (шифр для русских и английских букв)
4. Вывести на экран зашифрованное сообщение

Реализовать интерфейс пользователя с помощью WindowBuilder.

**Занятие 4**

**Построение графиков функций**

JFreeChart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart's extensive feature set includes:

* a consistent and well-documented API, supporting a wide range of chart types;
* a flexible design that is easy to extend, and targets both server-side and client-side applications (It is designed for use in applications, applets, servlets and JSP.);
* support for many output types, including Swing and JavaFX components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);
* JFreeChart is open source or, more specifically, free software. It is distributed under the terms of the GNU Lesser General Public Licence (LGPL), which permits use in proprietary applications.

JFreeChart can generate pie charts, bar charts (regular and stacked, with an optional 3D-effect), line charts, scatter plots, time series charts (including moving averages, high-low-open-close charts and candlestick plots), Gantt charts, meter charts (dial, compass and thermometer), symbol charts, wind plots, combination charts and more.

Additional features include:

• data is accessible from any implementation of the defined interfaces;

• export to PNG and JPEG image file formats (or you can use Java’s ImageIO library to export

to any format supported by ImageIO);

• export to any format with a Graphics2D implementation including:

– PDF via OrsonPDF (http://www.object-refinery.com/orsonpdf/);

– SVG via JFreeSVG (http://www.jfree.org/jfreesvg/);

• tool tips;

• interactive zooming (drag region and/or mouse-wheel) and panning;

• chart mouse events (these can be used for drill-down charts or information pop-ups);

• HTML image map generation;

• works in applications, servlets, JSP (thanks to the Cewolf project1) and applets;

JFreeChart is written entirely in Java, and should run on any implementation of the Java 2 platform (JDK 1.6.0 or later). Starting with JFreeChart 1.0.18, JavaFX support is available—this requires JDK 1.8.0 or later.

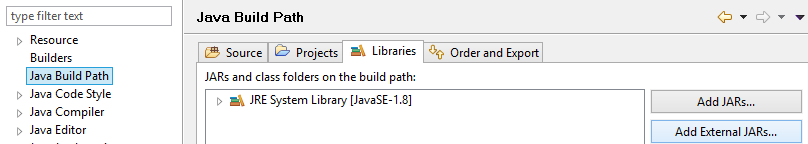
Скачать библиотеки, исходники, примеры и документацию:

<https://sourceforge.net/projects/jfreechart/files/2.%20Documentation/1.0.19/>

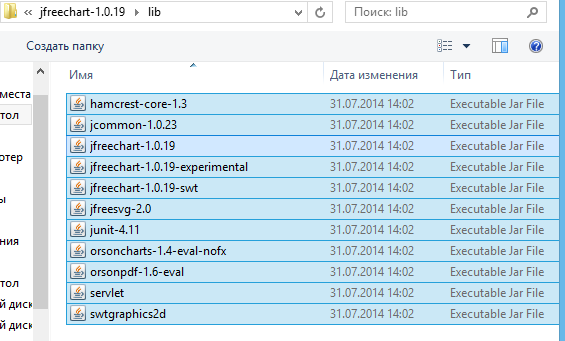
Разархивировать в удобном месте на компьютере.

**Как подключить дополнительные библиотеки:**

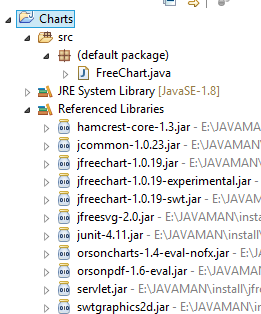
**ПК по проекту – Properties**



**Add External JARS – выделить разархивированные библиотеки**



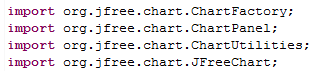
Результат:



Появились дополнительные библиотеки

**Пример 1 – Круговая диаграмма**

*Импорт объектов из дополнительных библиотек:*

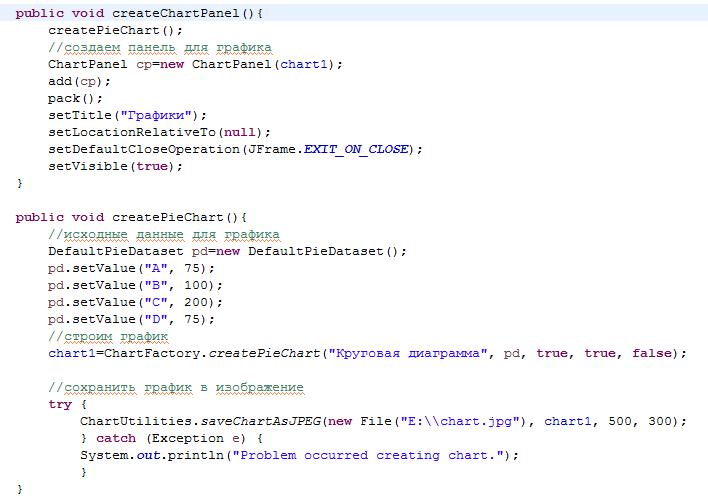


Создаем класс-фрейм.

Каждая созданная диаграмма имеет тип JFreeChart.



Для размещения графика служит специальная панель ChartPanel.

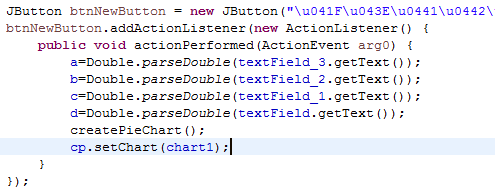


**Практическая работа**

С помощью WindowBuilder создать графический интерфейс для построения круговой диаграммы из 4-х сегментов:



*Примечание: двойной щелчок по кнопке добавляет к ней слушатель событий.*



Занятие 5

Практическая работа

Разработаем графический дизайн для ввода информации о слежении за объектом.

**Системы контроля версий. Github**

СКВ дает возможность сохранять версии своей программы и в любой момент возвращаться к любой из них.

Понятие: откат изменений (аналогия с ctrl+Z, только для сохраненных проектов).

Коммитить – сохранять версии программы.

Если в проекте изменялся только один файл, то СКВ сохраняет этот файл, а не весь проект (экономия памяти).

Централизованные системы управления версиями хранят все изменения на главном сервере (устарело). Новое поколение распределенных систем: серверные локальные ресурсы зеркалируются.

СЕРВЕР

save1

save2

save3

Разработчик2

save1

save2

save3

Разработчик1

save1

save2

save3

Сохранить

Сохранить

Рабочий процесс состоит из 3-х шагов:

Каталог Git (репозиторий) на сервере

Рабочий каталог на компьютере разработчика

Область подготовки файлов

3. Работа с версиями

1. Подготовить сохраненные файлы (сделать слепки)

2. Слепки поместить в каталог Git (на сервер)

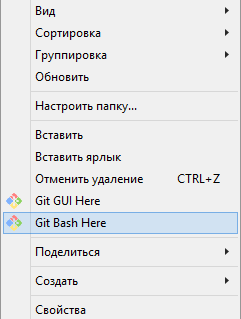
**Установка ПО**

Скачать <https://git-for-windows.github.io/>

Установить с настройками по умолчанию.

После установки можно работать как из командной строки, так и с GUI.

В контекстном меню появятся новые пункты:



**Работа в командной строке**

2. add .

(проиндексировать)

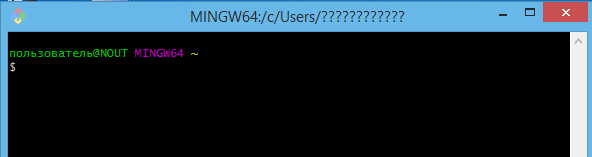
1. init

(создать репозиторий)

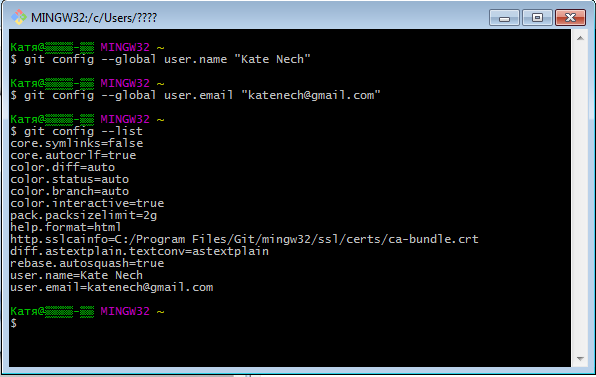
3. commit

(создать слепок, снимок)

Щелкнуть GitBash: 



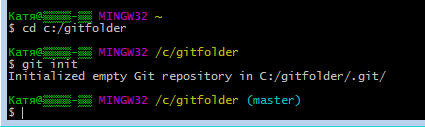
Можно (и даже нужно) ввести свой ник и адрес электронной почты:



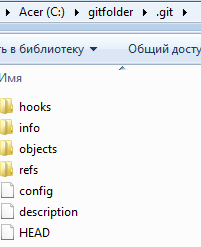
**Работа с GitBash**

1. Создать новый репозиторий (папку) **gitfolder** на своем компьютере:

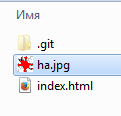
Создаем новый:



После инициализации в папке появляется скрытая папка .git. Содержимое:

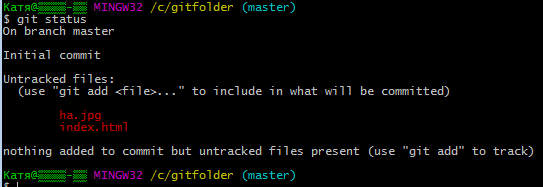


Поместим в репозиторий любые файлы:



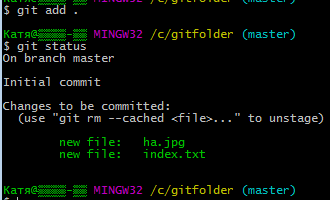
Пока это просто файлы, они не подлежат контролю версий.

Статус **status** репозитория:



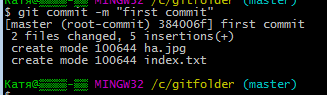
**2) Индексация**

Индексируем все командой **add .**



**3) Создание «слепка» (commit)**

После индексирования всех файлов можно делать коммит **commit**:



Статус:



**На этом подготовительная работа на локальном компьютере завершена.**

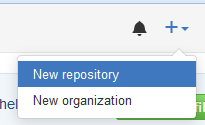
**Работаем с Github**

Заливаем проект на удаленный репозиторий – на данный момент самым популярным является Github.

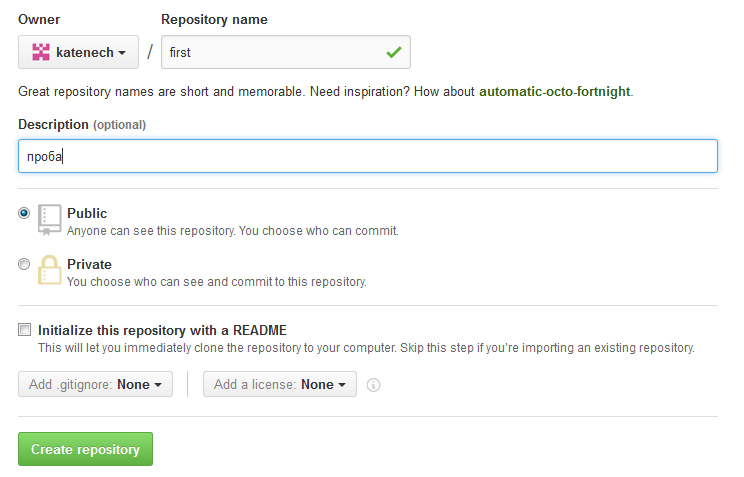
<https://github.com/> Зарегистрироваться.

**ШАГ 1**

Создаем новый репозиторий:



Public - любой может видеть ваши файлы (Private – платный).



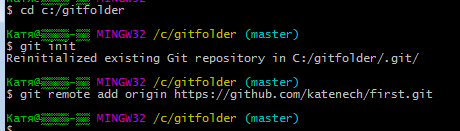
Нажать **Создать репозиторий**.

Копируем строку из раздела

### …or push an existing repository from the command line

git remote add origin <https://github.com/katenech/first.git>

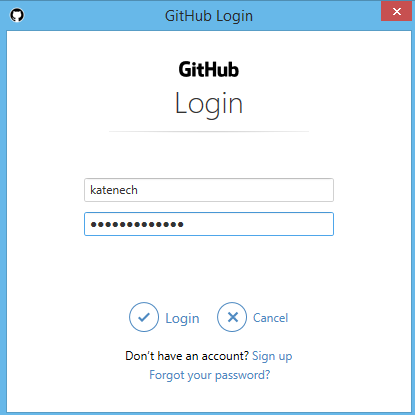
и вставляем ее в GitBash:



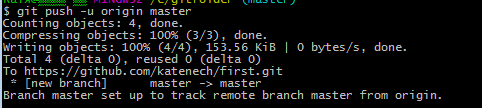
Подключаемся к Github

**git push -u origin master**

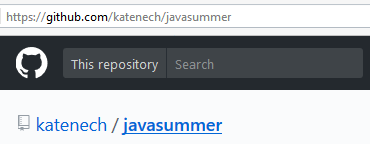
Вводим логин и пароль:

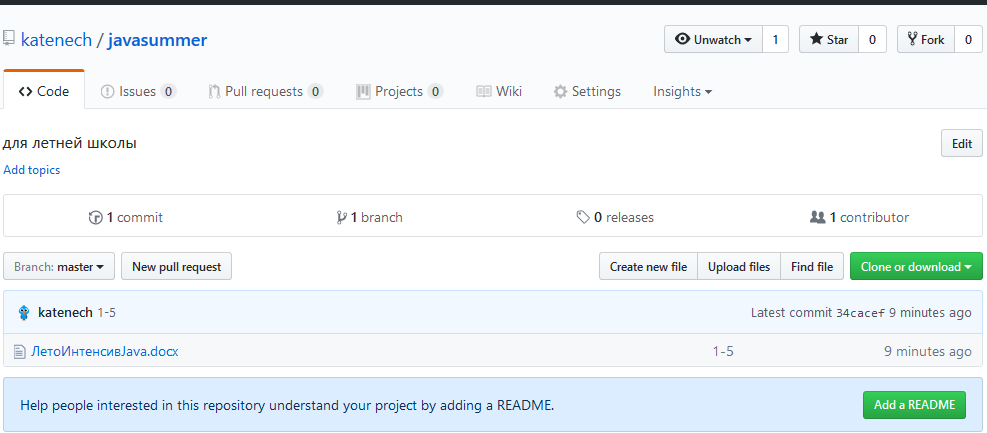


Результат:



Репозиторий на сайте (переходим по ссылке):





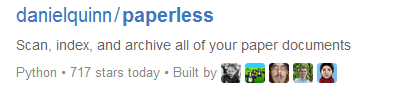
Можно просматривать файлы.

Можно увидеть все коммиты (версии, слепки, снимки), которые были сделаны в локальном репозитории.

Добавляю сегодняшний конспект в файл и выполняю все действия повторно. Результат:

**Клонируем сторонний репозиторий**

Выбираем популярные репозитории. Например:



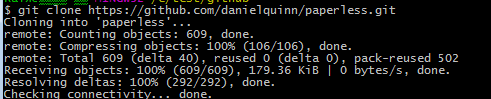
Клонирование по HTTPS:



Создадим папку на своем компьютере C:\test\github



Команда clone



Результат:

