# Frameworks

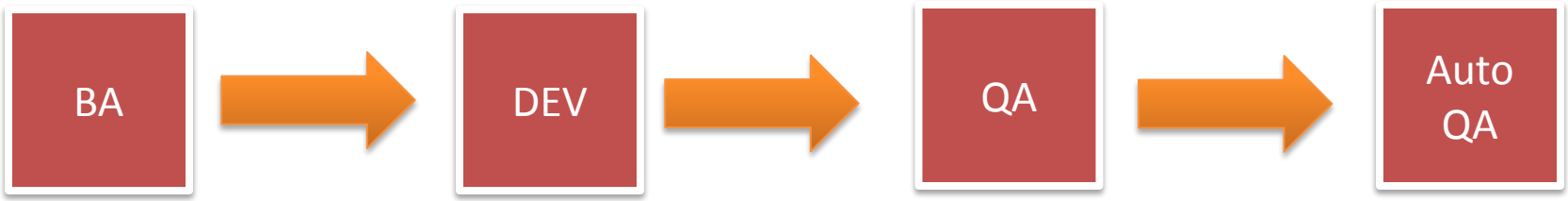
Игорь Хрол

# План

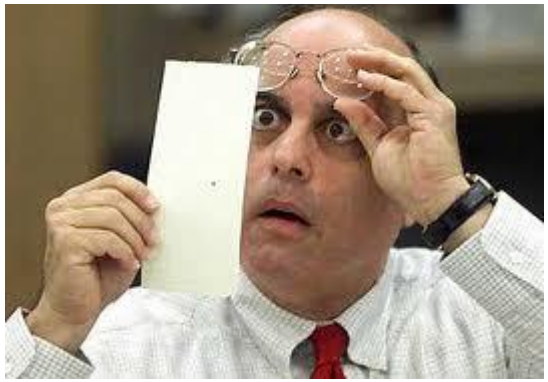
- Проблемы автоматизации
- Что такое фреймворк?
- Как он решает проблемы?
- Некоторые практики с фреймворка на Selenium RC

# Проблемы автоматизации

- В конце цикла разработки – много зависимостей



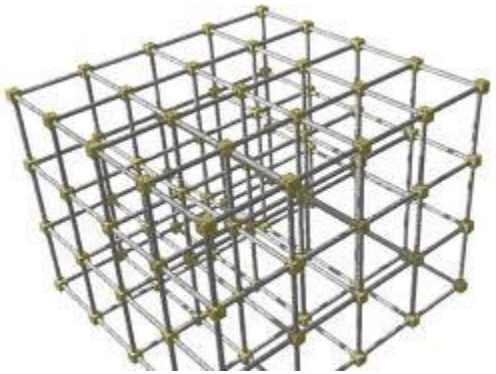

- Изменения UI
- Изменение бизнес-логики и тест-кейсов



# Что такое Framework?

- Каркас, структура, сфера деятельности
- В информационных системах структура программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

*Спасибо, Википедия*

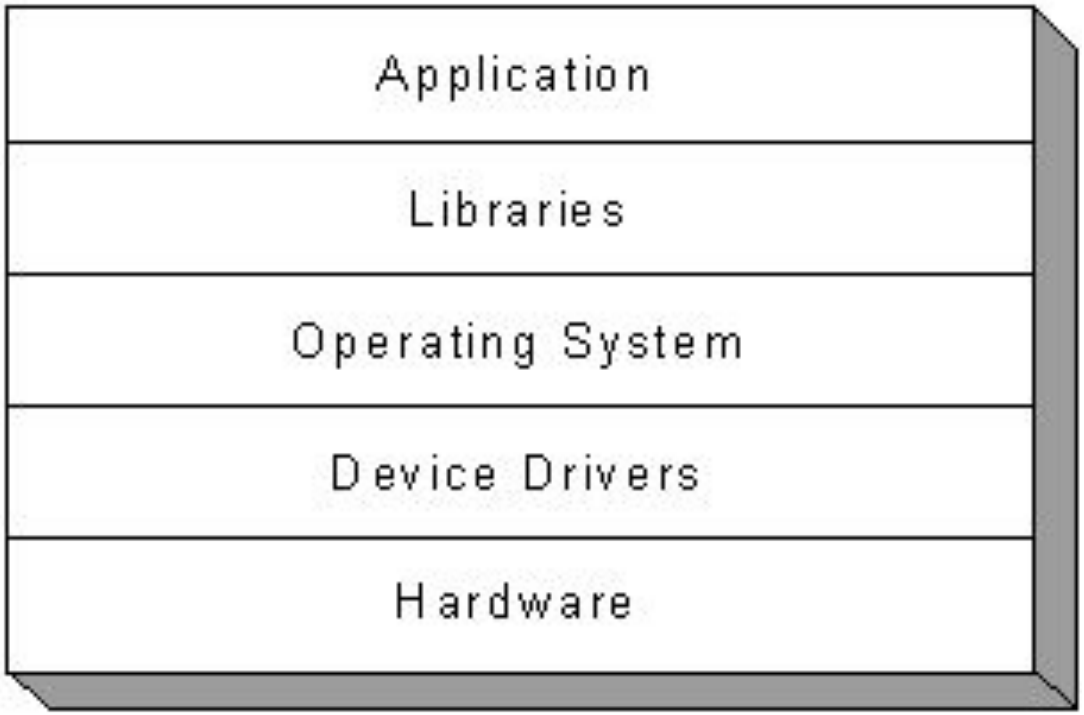


# Зачем нужен Framework?

- Ре-использование кода
  - Решение проблем одним изменением
  - Уменьшение количества изобретателей «велосипедов»

# Layering

- Подход к построению архитектуры систем

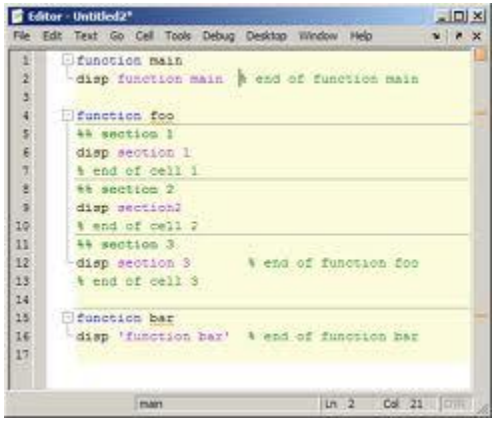


# UI Layer

- Цель: контролировать изменения UI
- Примеры:
  - Хранилище локаторов
  - Object Repository в QTP
  - Name Mapping в TestComplete
  - UI-Element в Selenium IDE

# Business components layer

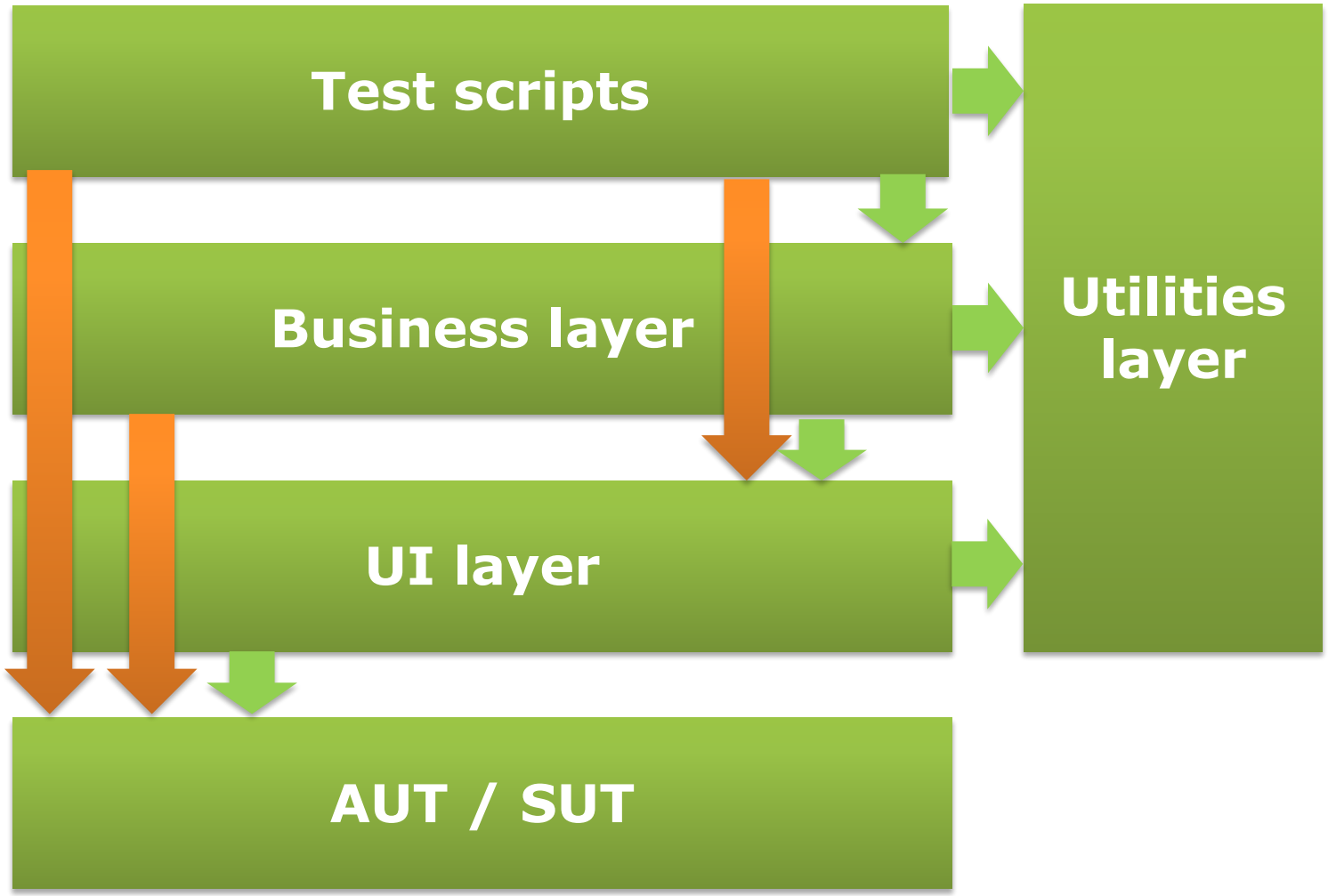
- Цель: контролировать изменения бизнес-логики
- Примеры:
  - Reusable Actions в QTP
  - Functions в TestComplete или любом другом инструменте



# Utilities

- Цель: вспомогательная
- Примеры:
  - Логгирование
  - Работа с базой данных
  - Чтение настроек и тестовых данных

# Типичная схема



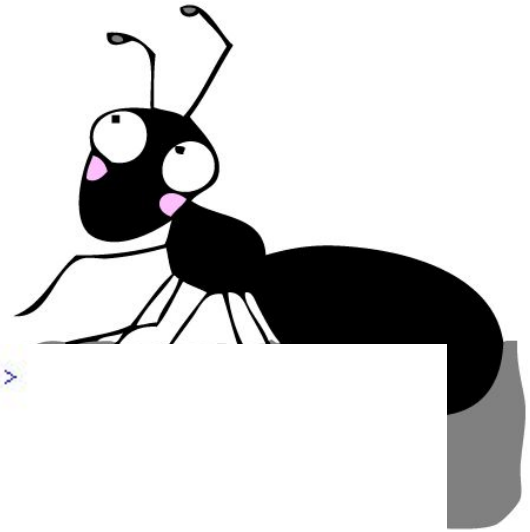
# KISS

- Используйте готовые решения
- Усложняйте архитектуру тогда, когда это нужно
- Старайтесь делать всё проще и очевидно

# Best Practices

# Запуск тестов из командной строки

- Ant (http://ant.apache.org/)
  - Запуск selenium server
  - Компиляция исходных кодов



```
<target name="selenium-start" description="Start the Selenium server">
   <exec spawn="true" executable="cmd.exe" >
       <arg value="/c"/>
       <arg value="java"/>
       <arg value="-jar"/>
       <arg value="${lib}/selenium-server.jar"/>
       <arg value="-userExtensions"/>
       <arg value="${lib}/user-extensions.js"/>
       <arg value="-port" />
       <arg value="${selenium.port}" />
       <arg value=">" />
       <arg value="${reports}/selenium_logs.logs" />
   </exec>
</target>

<target name="selenium-stop" description="Stop the Selenium server">
   <get taskname="selenium-shutdown"
       src="http://localhost:${selenium.port}/selenium-server/driver/?cmd=shutDownSeleniumServer"
       dest="${reports}/selenium-server-shutdown.txt" ignoreerrors="true" />
</target>
```

# Как хранить общий доступ к selenium?

```java
package com.epam.seleniumwebinar.core;

import com.thoughtworks.selenium.DefaultSelenium;

public class WebinarSelenium extends DefaultSelenium {

    public WebinarSelenium(String serverHost, int serverPort,
            String browserStartCommand, String browserURL) {
        super(serverHost, serverPort, browserStartCommand, browserURL);
    }

    public void clickAndWait(String locator) {
        this.click(locator);
        this.waitForPageToLoad("30000"); //TODO update to read from properties
    }

    public static WebinarSelenium selenium;

}
```

для

```java
import static com.epam.seleniumwebinar.core.WebinarSelenium.selenium;

public class NewTest {

  @Test
  public void f() {
      selenium.open("http://www.google.com");
      selenium.type("q", "Ищу человека");
      selenium.clickAndWait("btnG");
      Assert.assertTrue(selenium.isTextPresent("человек"), "Не нашёл человека");
      Reporter.log("Test passed");
  }
```

# Базовый класс для всех тест кейсов

NewTest.java | LoginForm.java | ILoginForm.java | WebinarSelenium.java | BaseTest.java

```
package com.epam.seleniumwebinar.core;

import static com.epam.seleniumwebinar.core.WebinarSelenium.selenium;

import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;

public class BaseTest {

    @BeforeClass
    public void beforeClass() {
        selenium = new WebinarSelenium("localhost", 4444, "*firefox",
                "http://example.com/");
        selenium.start();
    }

    @AfterClass
    public void afterClass() {
        selenium.stop();
    }
```

```
import com.epam.seleniumwebinar.core.BaseTest;

public class NewTest extends BaseTest {

  @Test
  public void f() {
      selenium.open("http://www.google.com");
      selenium.type("q", "Ищу человека");
      selenium.clickAndWait("btnG");
      Assert.assertTrue(selenium.isTextPresent("человек"), "Не нашёл человека");
      Reporter.log("Test passed");
  }

}
```

# Структура тест-кейса

- Preconditions
- Test case
- Cleanup

| Flow | Preconditions | Main | Cleanup |
|---|---|---|---|
| Successful | Passed | Passed | Passed |
| Failed cleanup | Passed | Passed | Failed |
| Failed test case | Passed | Failed | Passed/Failed |
| Failed preconditions | Failed | Not run | Passed/Failed |

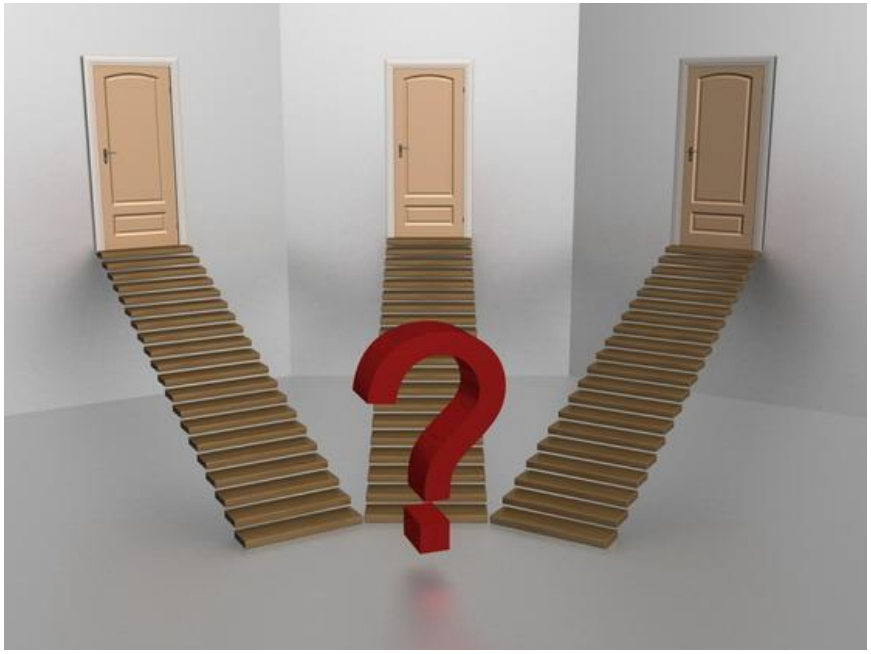
# Где хранить настройки?

- Найдите грань между настройками и тестовыми данными:
  - Selenium host
  - Selenium port
  - Timeout
  - Base URL
- properties-файлы – стандарт для Java (http://download.oracle.com/javase/1.4.2/docs/api/java/util/Properties.html)
- config.xml – для C#

# Где хранить тестовые данные?

- Excel
- CSV-файлы
- База данных
- XML

```
<?xml?>
```



# Excel-хранилища данных

- Табличные данные на каждом sheet
- Key-value пары в столбец
- Именованные диапазоны
- Чем читать excel?
  - JDBC (http://www.ehow.com/how_4810548_use-jdbc-connect-ms-excel.html)
  - POI (http://poi.apache.org/)

# Где хранить локаторы?

- Поля классов

# Где хранить локаторы?

- Поля интерфейсов

NewTest.java | Authentification.java | LoginForm.java | ILoginForm.java

```
package com.epam.seleniumwebinar.ui;

public interface ILoginForm {

    public String userField = "css=input#username";
    public String passwordField = "css=input#password";
    public String rememberMeCheckbox = "css=form#login input.checkable";

}
```

NewTest.java | Authentification.java | LoginForm.java | ILoginForm.java

```
package com.epam.seleniumwebinar.ui;

public class LoginForm implements ILoginForm {

    public void checkRememberMe() {
        //some code
    }

}
```

# Где хранить локаторы?

- properties-файлы

```
public void testNew() throws Exception {

        // Open app url.
        selenium.open("http://www.test.com");

        // Provide admin username.
        selenium.type(admin.username, "xxxxxxxx");

        // Click on Login button.
        selenium.click(admin.loginbutton);

        // Click on Create New Event button.
        selenium.click(admin.events.createnewevent);
        selenium.waitForPageToLoad("30000");
```

```
admin.username = loginForm:tbUsername
admin.loginbutton = loginForm:btnLogin
admin.events.createnewevent = adminHomeForm:_activitynew
admin.events.cancel = addEditEventForm:_IDcancel
admin.events.viewoldevents = adminHomeForm:_activityold
```

- http://seleniumhq.org/docs/06_test_design_considerations.html#ui-mapping

# Структура UI Layer

- Разбивка на страницы
- Сбор «особенных» UI элементов в отдельные классы:
  - Различные кастомизированные кнопки-поля (нужно вызывать специальные события)
  - Upload-download файлов
- Иерархия объектов (GWT, например)

# Структура Business Layer

- Набор статических методов, объединённых по классам

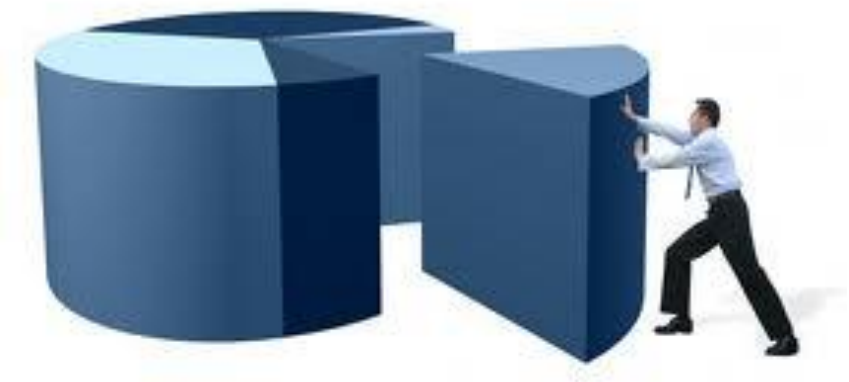
NewTest.java | Authentification.java

```
package com.epam.seleniumwebinar.business;

public class Authentification {

    public static void login(String username, String password) {
        //some code
    }

    public static void logout() {
        //some code
    }
}
```

- Чем проще, тем лучше

# Tracebility с ручным тестированием

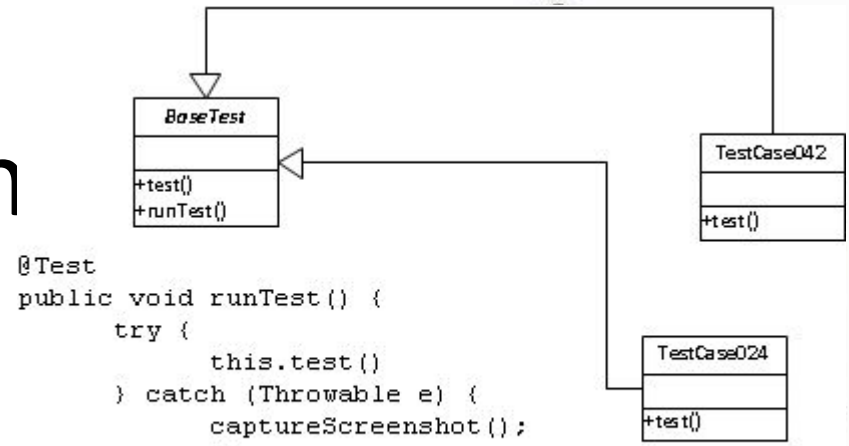
- Соотнесение тест-кейсов
- Соотнесение тестовых шагов
- Интеграция с существующей отчётностью
  - HP Quality Center
  - TestLink
  - JIRA



# Отчёты

- HTML
- Группировка по test suite и test case
- Screenshots
- Максимально user friendly
- При возможности – интеграция с существующей системой отчётности

# Screensh




```
@Test
public void runTest() {
      try {
            this.test()
      } catch (Throwable e) {
            captureScreenshot();
            throw e;
      }
}
```

- void **captureScreenshot** (St
- Обычно делается во время ошибки
  - Try-catch в BaseTest
  - Listener в JUnit
- http://www.cloudtesting.com/blog/2009/06/24/capturing-screen-shots-of-browsers-with-selenium-and-cloud-testing-part-1/

# Вопросы?