

Справочник по работе с сервлетами (IntelliJ IDEA)

Существует множество интегрированных сред разработки (ИСР) подходящих для наших задач, но наиболее популярные – это *Eclipse* и *IntelliJ IDEA* от *JetBrains*. Мы рассмотрим вторую.

Для скачивания ИСР *IntelliJ IDEA* нужно перейти по данной ссылке: <https://www.jetbrains.com/idea/download/>

Для работы с сервлетами необходимо скачать **Ultimate** версию. Если вы студент или преподаватель, то вы можете получить ее бесплатно, подробности можно найти здесь:

<http://confluence.newit.gsu.by/display/dldoc/Student+Developer+Pack>

После скачивания запускаем скачанный исполняемый файл. Открывается окно, которое показано на рисунке 1.

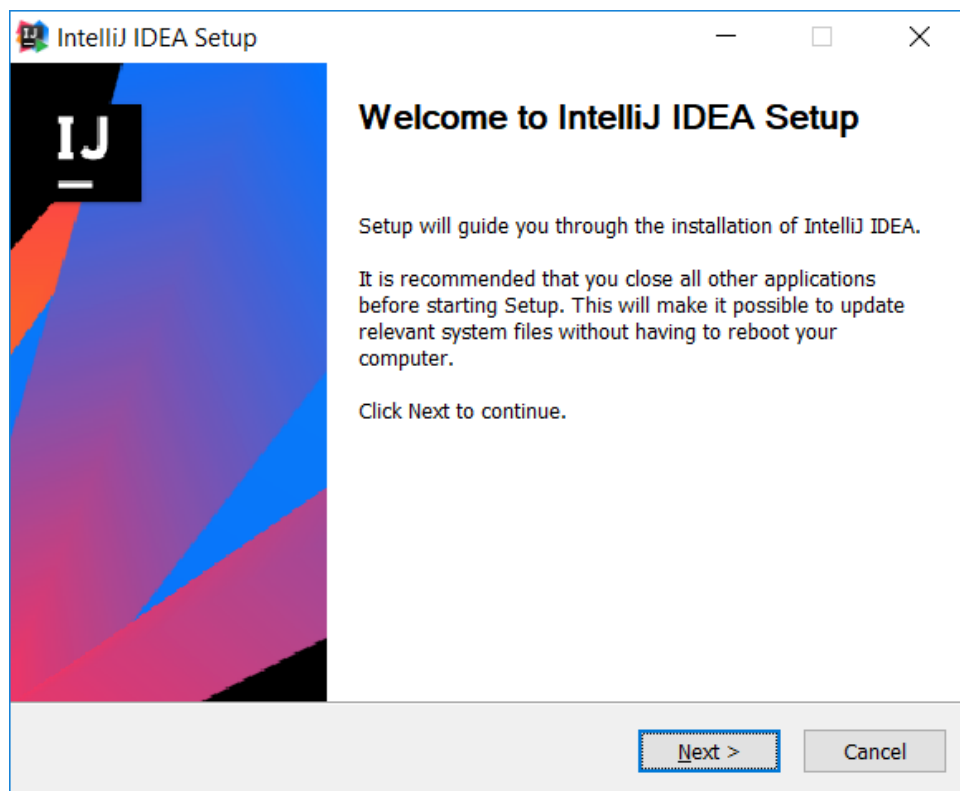


Рисунок 1 – Начало установки

Выбираем путь установки (рисунок 2).

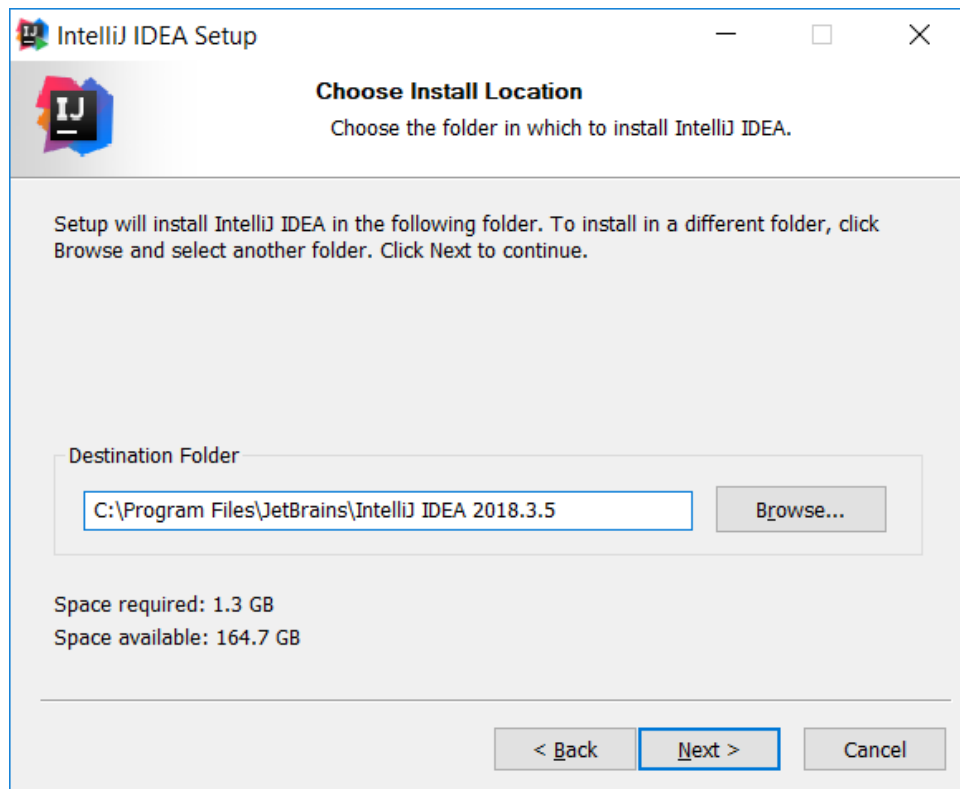


Рисунок 2 – Выбор установочного пути

Выбираем дополнительные параметры, как показано на рисунке 3.

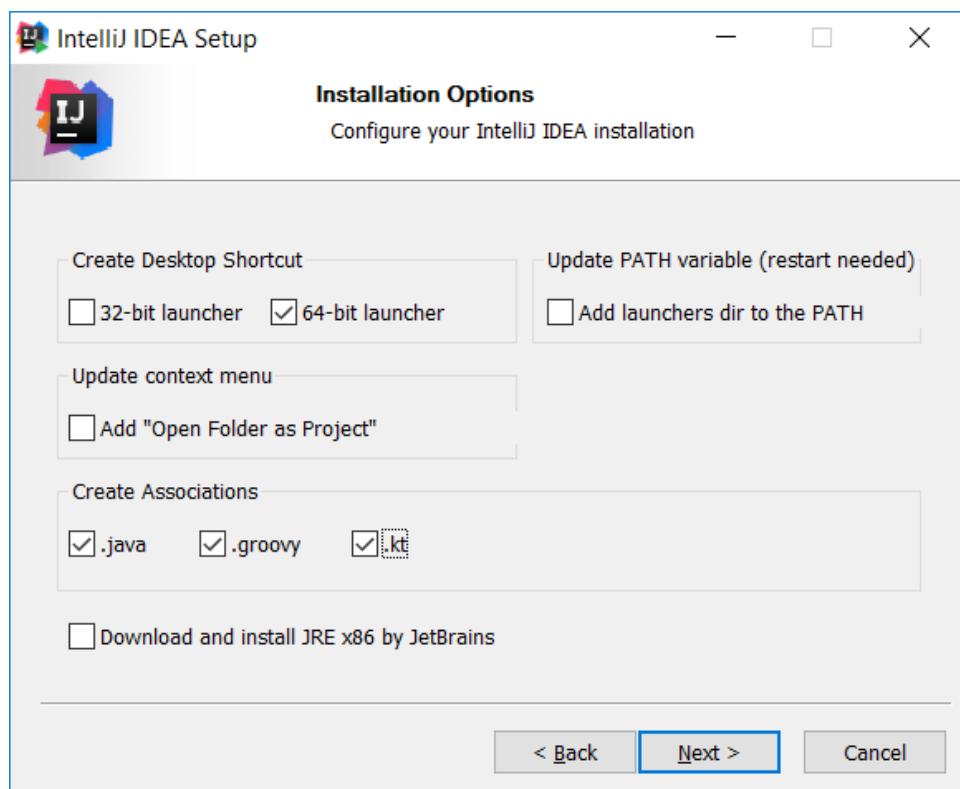


Рисунок 3 – Выбор дополнительных параметров

Устанавливаем ИСР (рисунок 4).

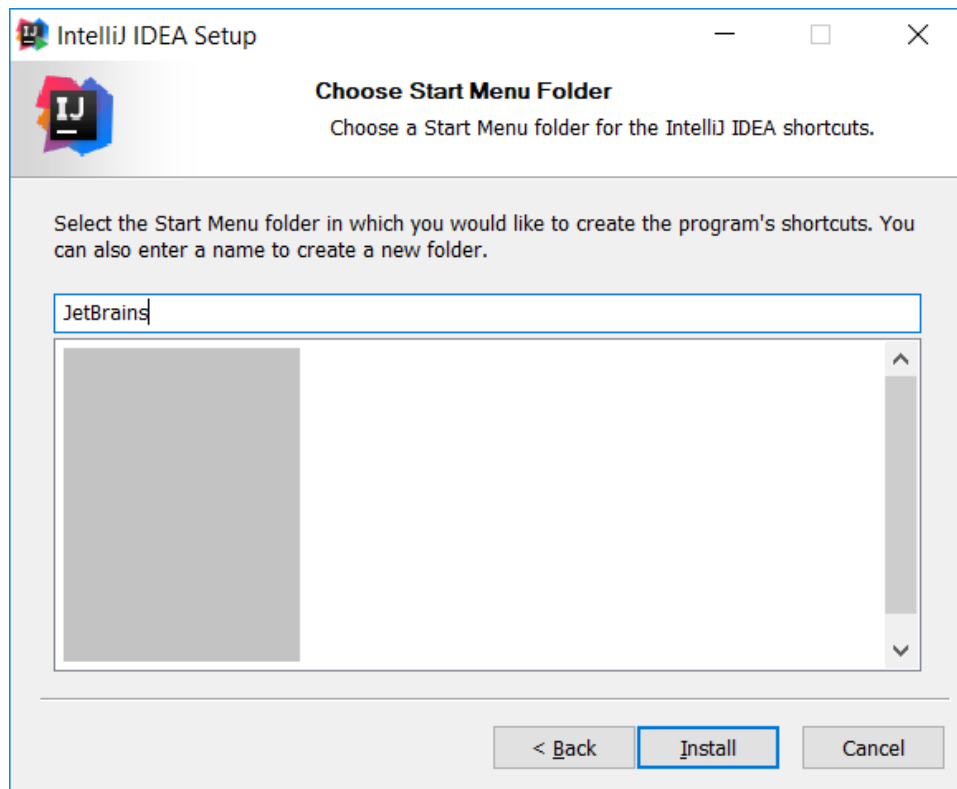


Рисунок 4 – Установка

После установки появится окно, как на рисунке 5. Устанавливаем флажок для запуска *IntelliJ IDEA* и нажимаем *Finish*.

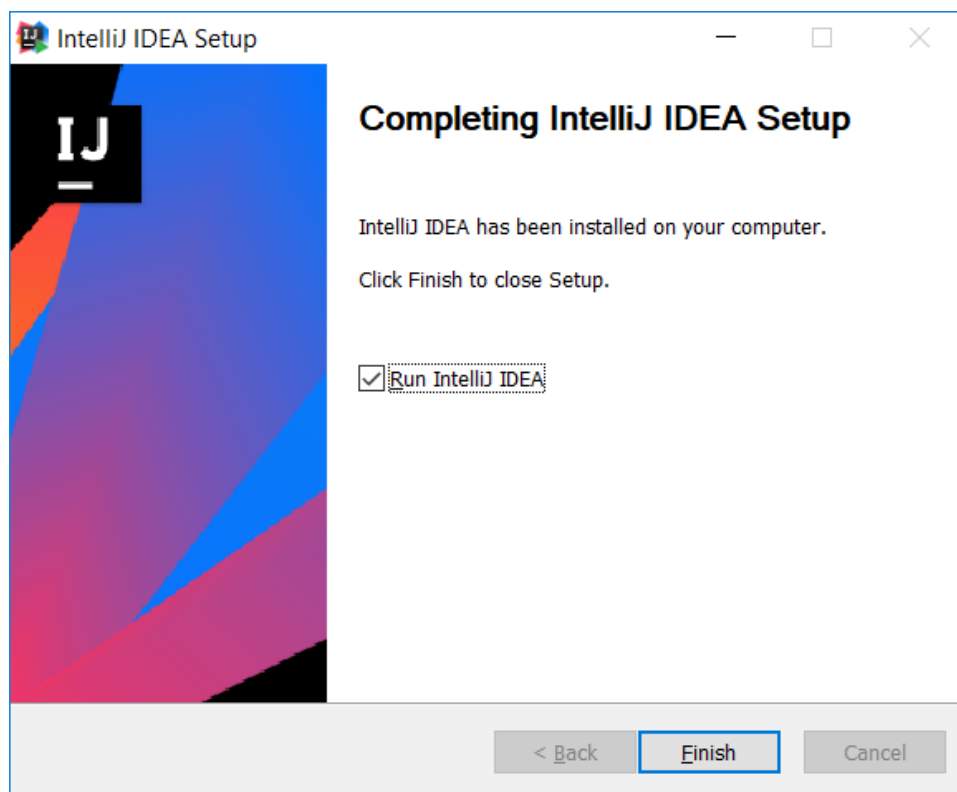


Рисунок 5 – Запуск

Предлагается импортировать настройки (рисунок 6). Мы устанавливаем впервые, следовательно, отказываемся.

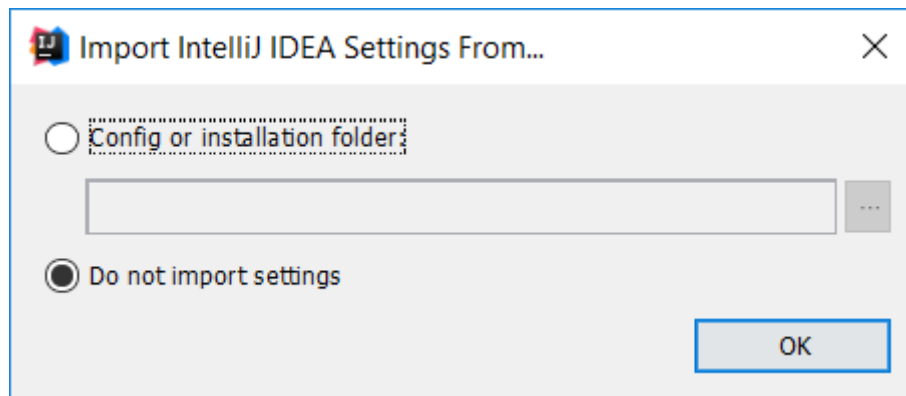


Рисунок 6 – Импорт настроек

Подтверждаем лицензионное соглашение (рисунок 7).

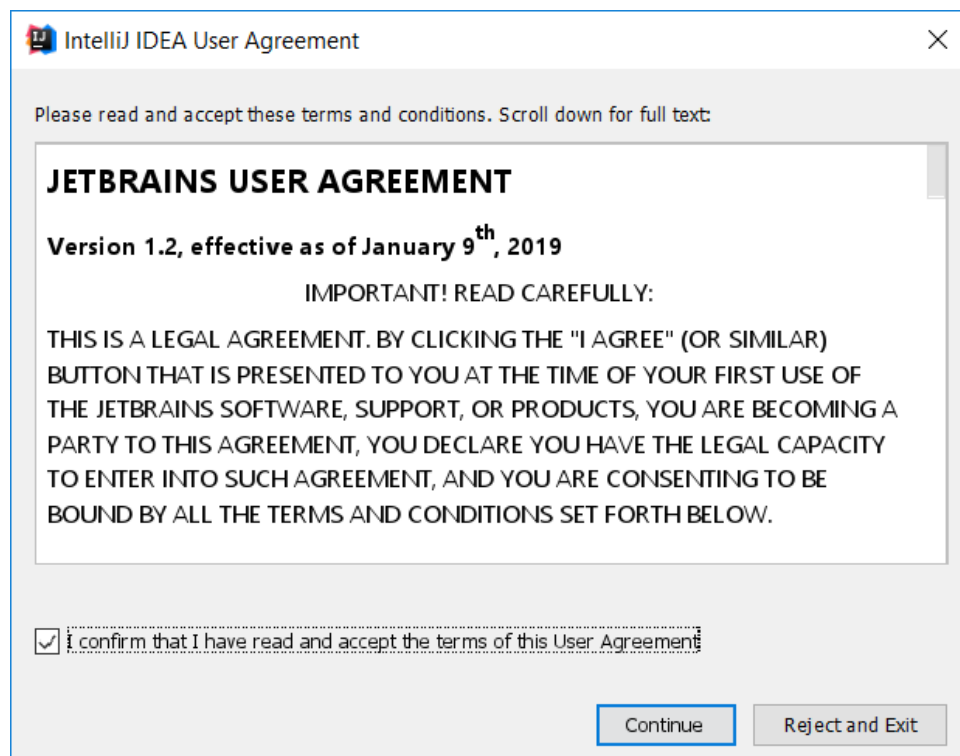


Рисунок 7 – Лицензионное соглашение

На рисунке 8 нажимаем *Don't send*, что позволит не отправлять данные и статистику использования *IntelliJ IDEA*.

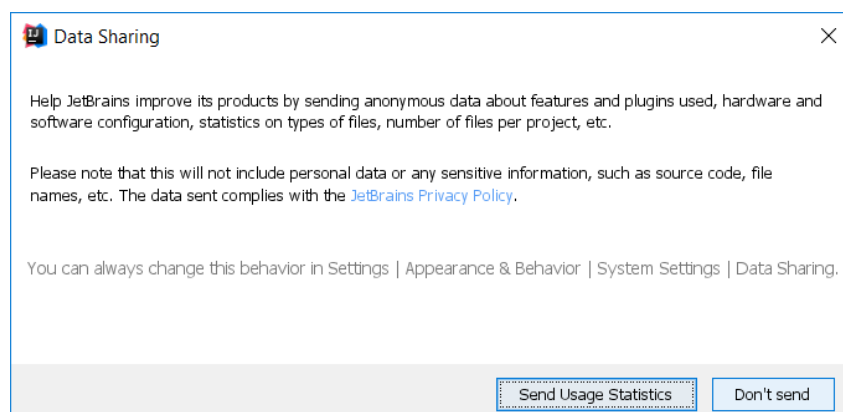


Рисунок 8 – Сбор статистики

Выбираем цветовую схему и нажимаем *Skip Remaining and Set Defaults* (рисунок 9).

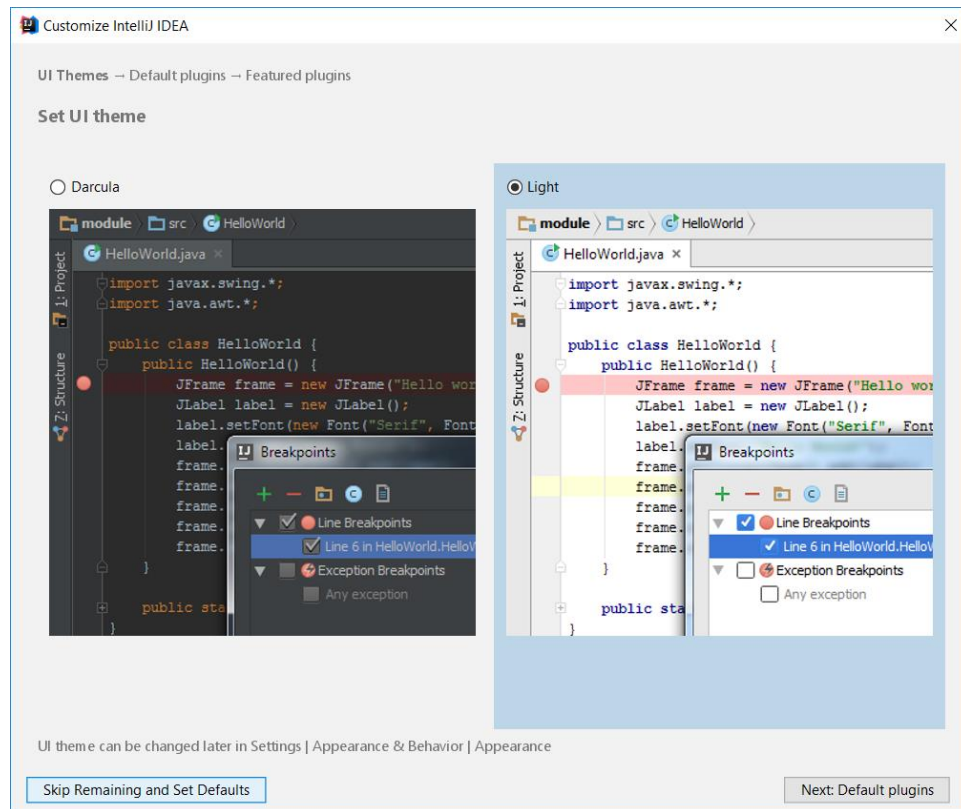


Рисунок 9 – Выбор цветовой схемы

Открылось окно активации. Вводим аккаунт от *JetBrains* (рисунок 10). Также можно просто ввести код активации (его можно найти в личном кабинете на сайте *JetBrains*).

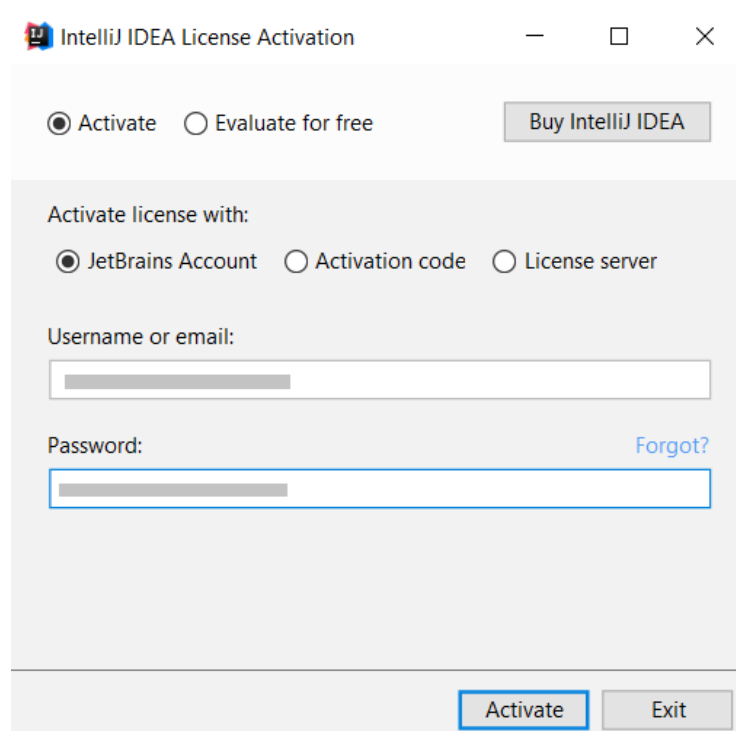


Рисунок 10 – Активация

Появляется окно, где нажимаем *Create New Project* (рисунок 11).

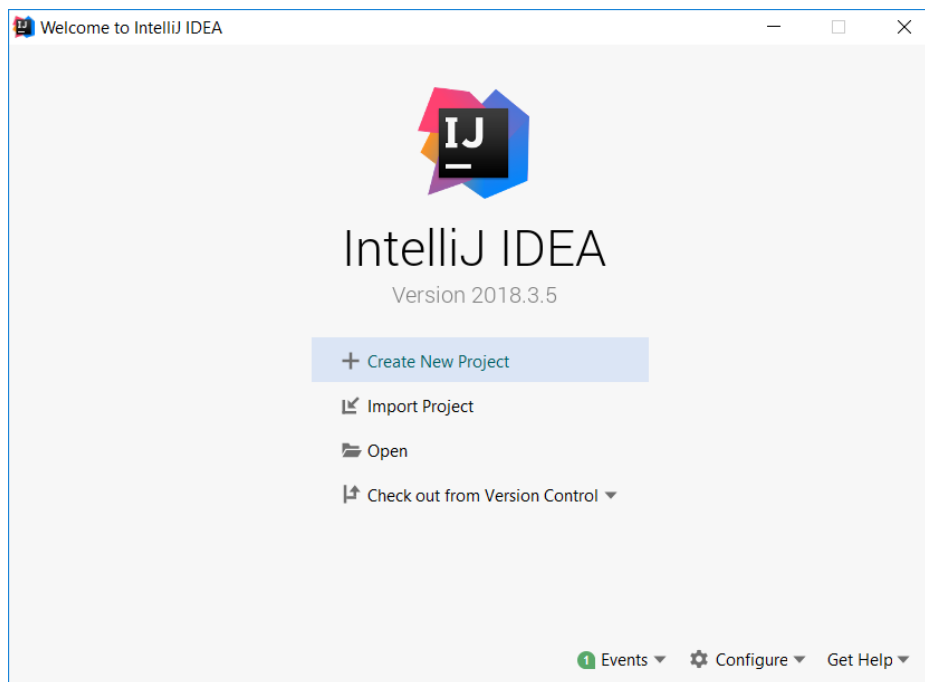


Рисунок 11 – Создание проекта

Перед созданием проекта нам нужно скачать контейнер сервлетов *Apache Tomcat* (иначе говоря – веб-сервер). Это можно сделать по ссылке: <https://tomcat.apache.org/download-90.cgi>

Скачиваем архив в зависимости от операционной системы (рисунок 12).

- Core:
 - [zip](#) ([pgp](#), [sha512](#))
 - [tar.gz](#) ([pgp](#), [sha512](#))
 - [32-bit Windows zip](#) ([pgp](#), [sha512](#))
 - [64-bit Windows zip](#) ([pgp](#), [sha512](#))

Рисунок 12 – Загрузчики Apache Tomcat

Распаковываем данный архив в любое место. В итоге получится примерно так: *C:\apache-tomcat-9.0.13*.

Возвращаемся к созданию проекта. Выбираем тип проекта – *Java Enterprise* (также ставим галочку напротив *Web Application* и *Create web.xml*; последнее позволит сгенерировать *web.xml* файл, информация об этом файле представлена в конце справочника). В поле *Project SDK* выбираем установленный *JDK*, а в поле *Application Server* – *Apache Tomcat*, который мы скачали выше. Нажимаем *Next* (рисунок 13). Если у вас нет пункта *Java Enterprise*, то, скорее всего, вы установили *Community* версию, а не *Ultimate*.

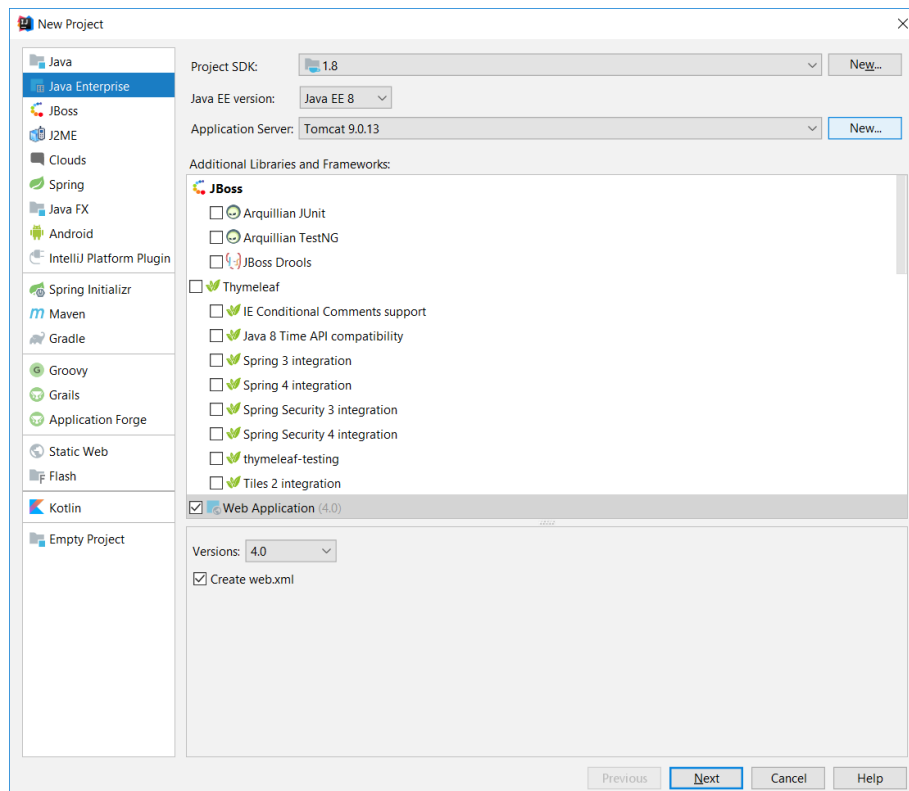


Рисунок 13 – Создание проекта

В поле *Project name* вводим имя нашего проекта. Создаем каталог (в данном случае *Checkboxes*) и указываем к нему путь. Нажимаем *Finish* (рисунок 14).

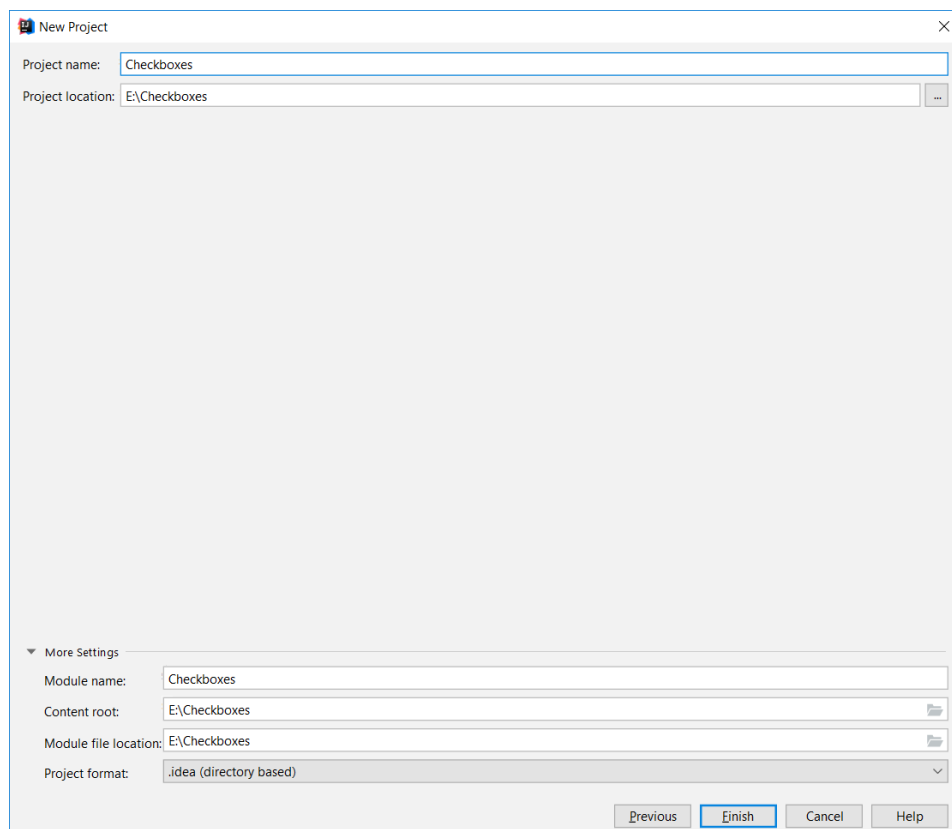


Рисунок 14 – Именованние проекта

Итак, у нас появилась страница, как на рисунке 15.

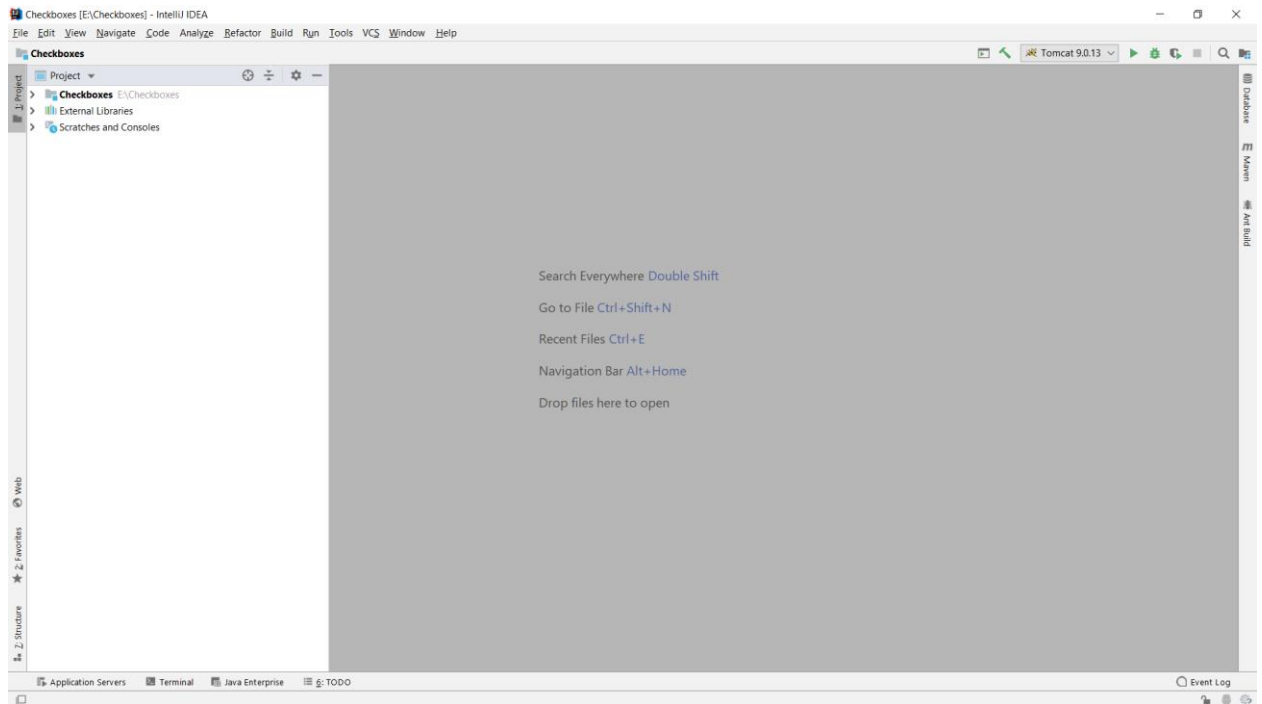


Рисунок 15 – Стартовая страница

Рассмотрим простую задачу для демонстрации запуска сервлета (рисунок 16).

Решение 20 Обсудить задачу в форуме (-): Korotchikov Nikita \Servlet\1 - "Проверить чекбоксы" 219082 Korotchikov Nikita, HTTP-31, февраль 2019

Баллов: 10
Решили: 0
Сложность: 1

Условие

Дана html-страница (см. пример). На ней находится список предметов. Пользователь, заходя на страницу, должен выбрать список предметов (может и не выбрать), которым он обучается, а затем отправить запрос на сервер. Итак, ваша задача состоит в том, чтобы определить (вывести) список предметов (значения которых содержатся в атрибуте value тега input), которые отметил пользователь. Каждый предмет, который будет выведен, нужно поместить в тег <p>.

Пример

Например, дана html-страница:

Checkboxes x +

localhost:8080/checkboxes.html

☒ Математика ☐ Физика ☒ Информатика

Если пользователь нажмет на кнопку отправить, то он получит следующее (обратите внимание на URL):

localhost:8080/Servlet?subject=

localhost:8080/Servlet?subject=subject1&subject=subject3

subject1

subject3

Информация

Расширение файла-решения	java
Метод передачи параметров	GET
Атрибуты URL-запроса	subject
Подсказки	Просмотр
Решение	Просмотр

Отправка решения:

Файл не выбран

Delta

Рисунок 16 – Задача

Обратите внимание, что решение данной задачи – сервлет (*.java-файл*). Поэтому создадим его, как показано на рисунке 17.

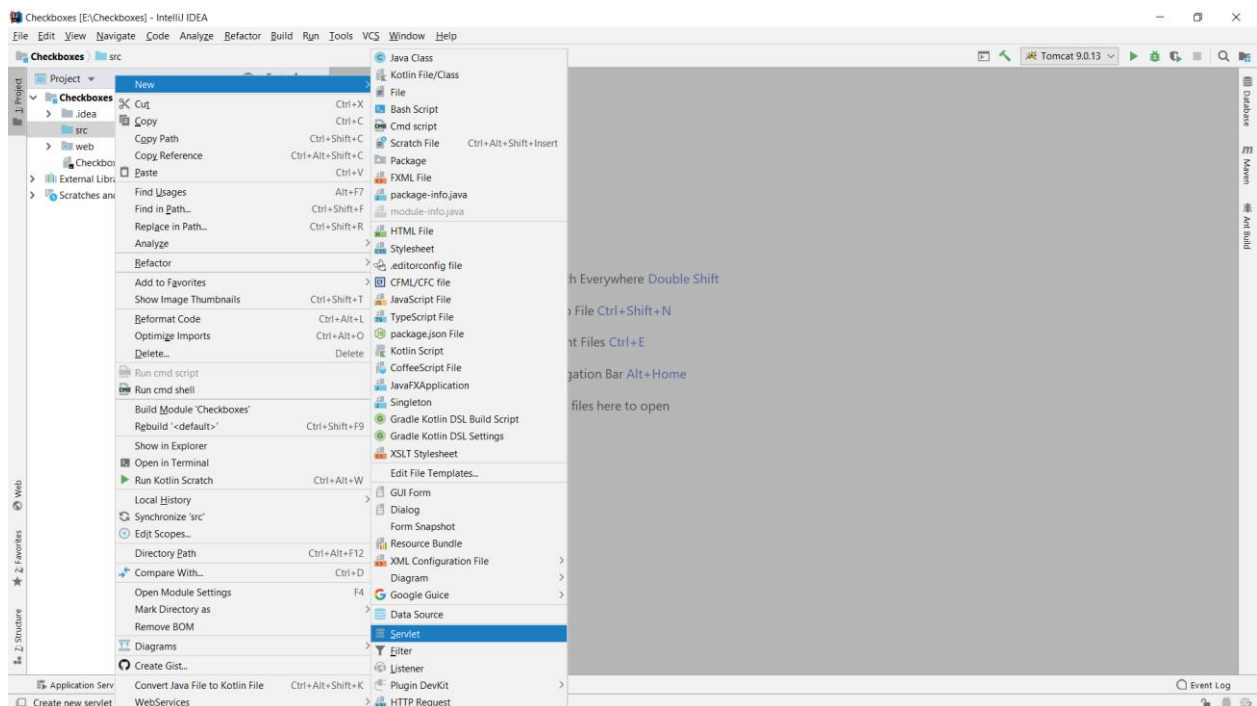


Рисунок 17 – Создание сервлета

Вводим название нашего сервлета и отмечаем *Create Java EE 6 annotated class*, что позволит нам обойтись без файла *web.xml* (информация об этом файле представлена в конце справочника). Нажимаем *OK* (рисунок 18).

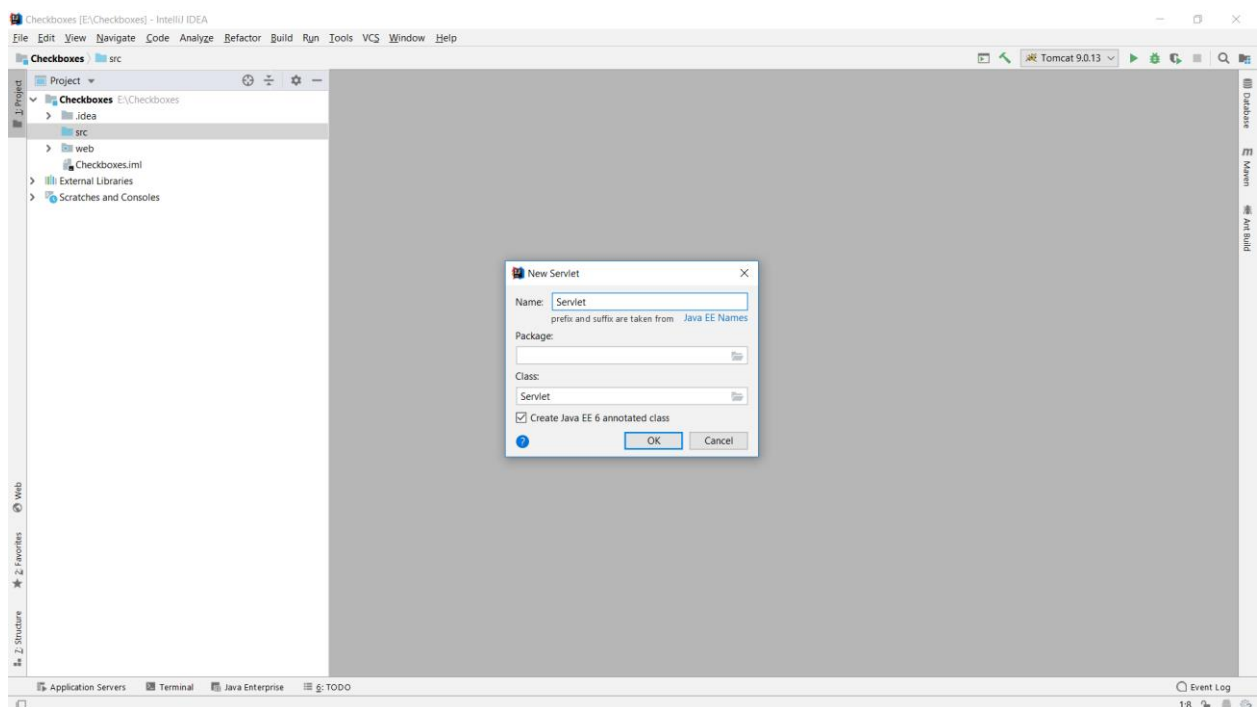


Рисунок 18 – Создание сервлета

Итак, созданный сервлет показан на рисунке 19.

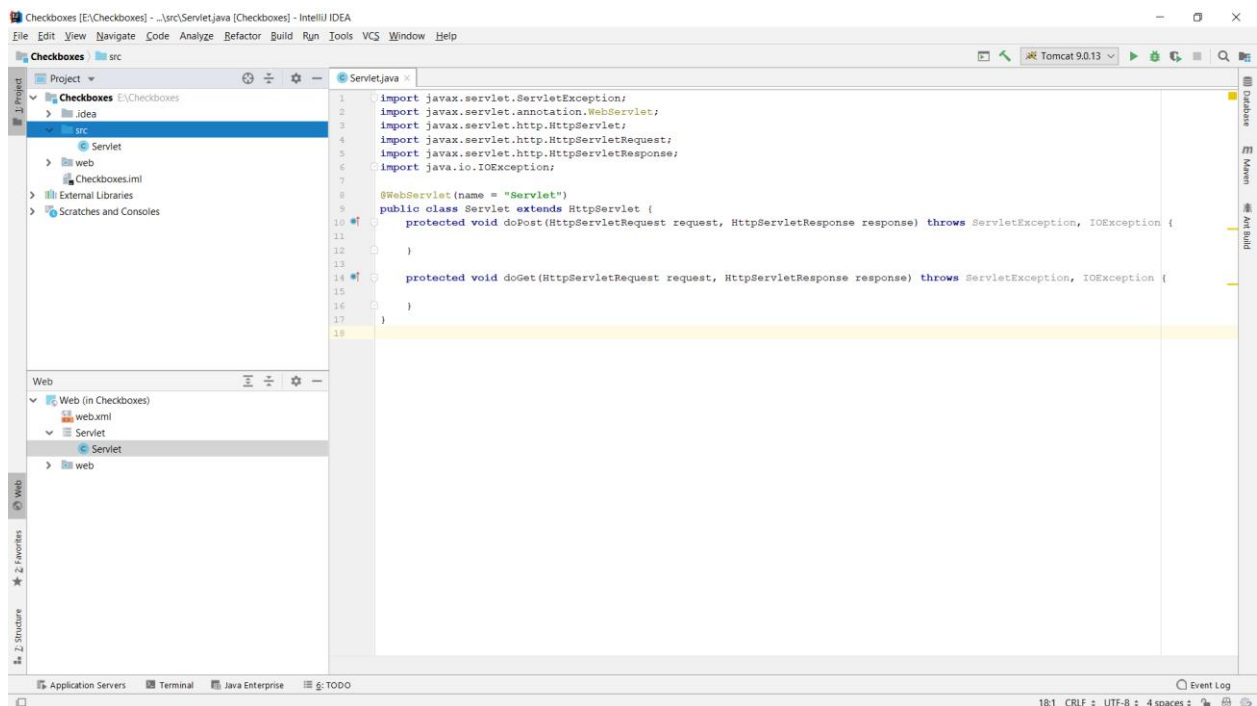


Рисунок 19 – Созданный сервлет

Так как это обучение по установке и запуску, то мы не будем акцентировать внимание на решении данной задачи. Оно представлено на рисунке 20.

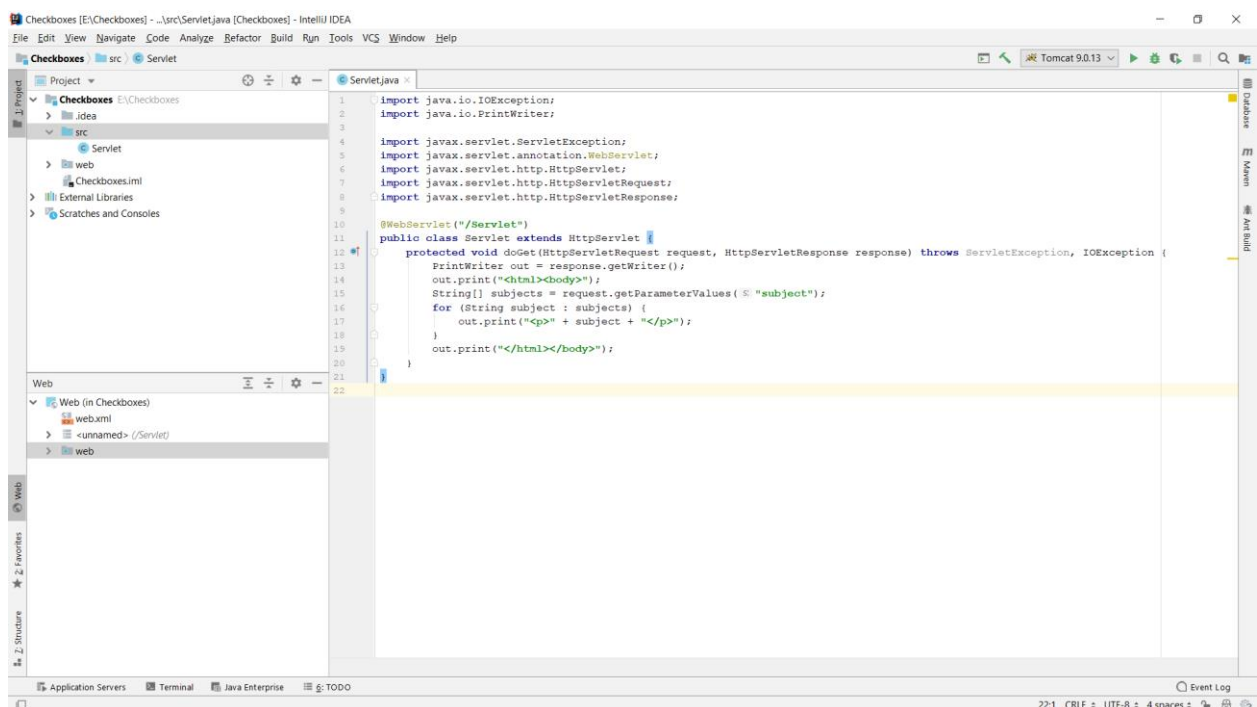


Рисунок 20 – Решение задачи

Но в задаче сказано, что имеется и исходный *HTML-файл*, где пользователь будет выбирать предметы и нажимать на кнопку отправить. Скачиваем его и помещаем в наш проект – каталог *web* (рисунок 21). Также важно заметить, что у тега *<form>* определены два атрибута: *action* и *method*. Атрибут *action* определяет адрес (в данном случае сервлет; где и как задается

адрес сервлета сказано в последнем разделе данного справочника: через файл *web.xml* или через аннотацию), куда будет отправлена форма, а атрибут *method* определяет метод передачи параметров запроса.

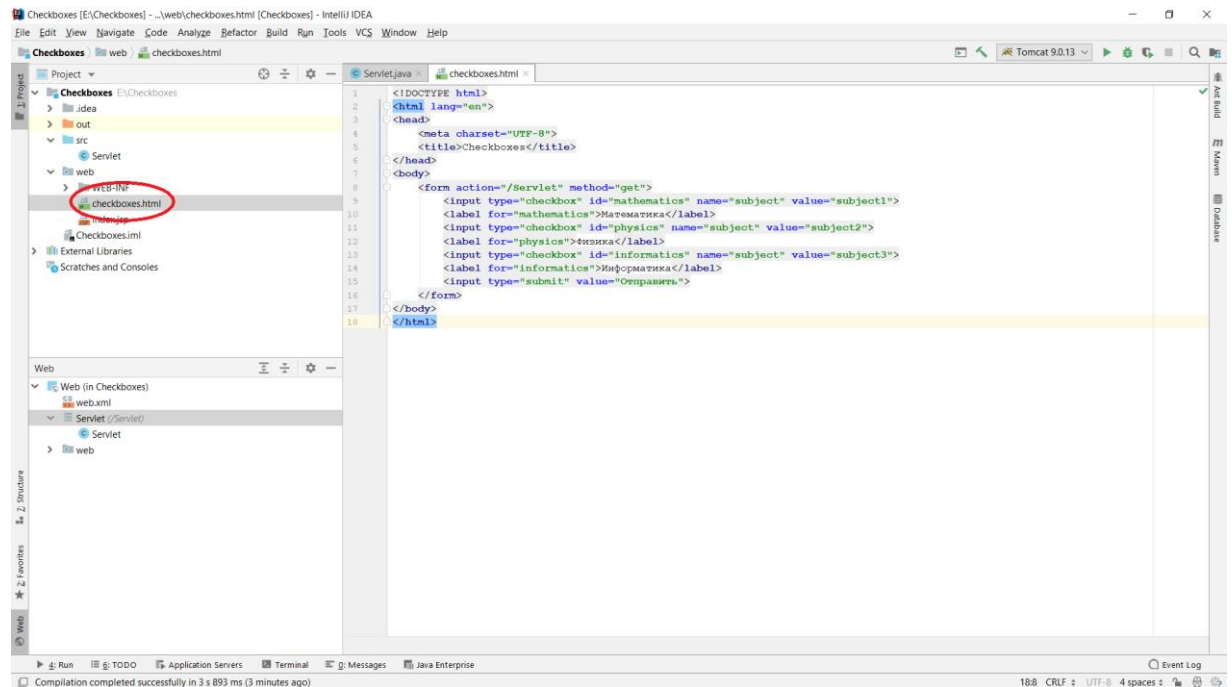


Рисунок 21 – Добавление HTML-файла

Запускаем проект (рисунок 22).

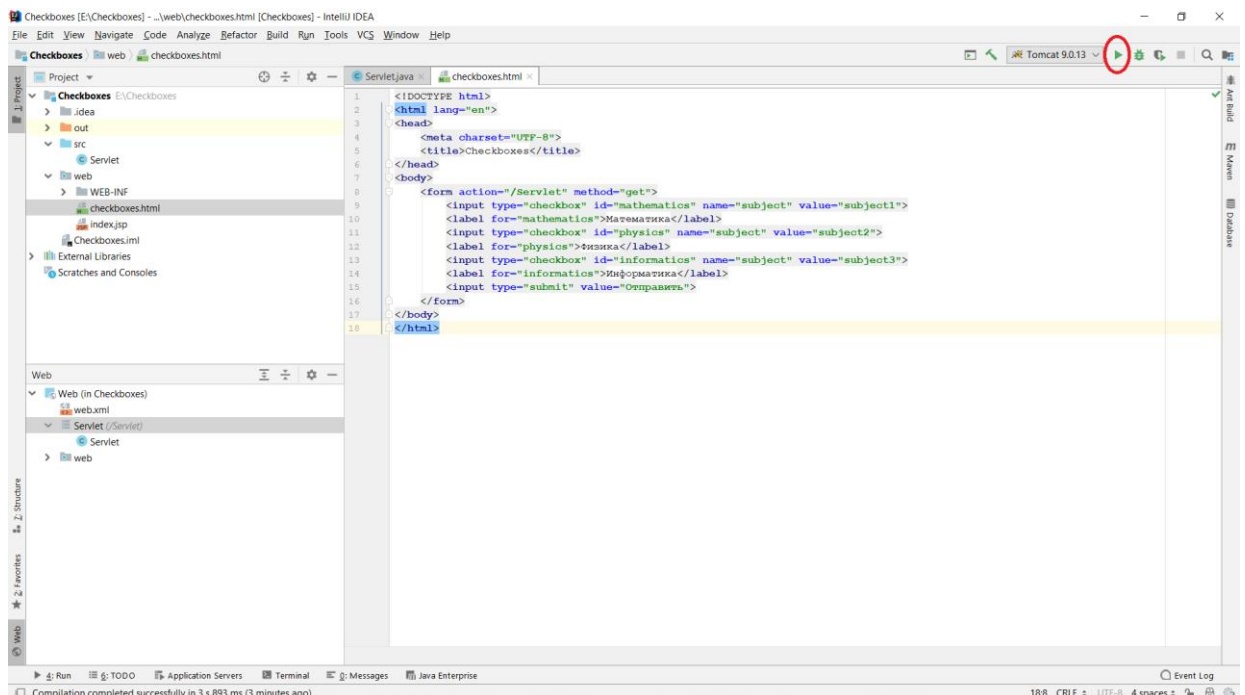


Рисунок 22 – Запуск проекта

После нескольких секунд ожидания автоматически откроется браузер. Вероятно, вы увидите запись *\$END\$*. Это открывается страница *index.jsp* при запуске, поэтому введите *URL* самостоятельно, как показано на рисунке 23.

Результаты работы *HTML-страницы* и сервлета показаны на рисунках 23 и 24 соответственно.

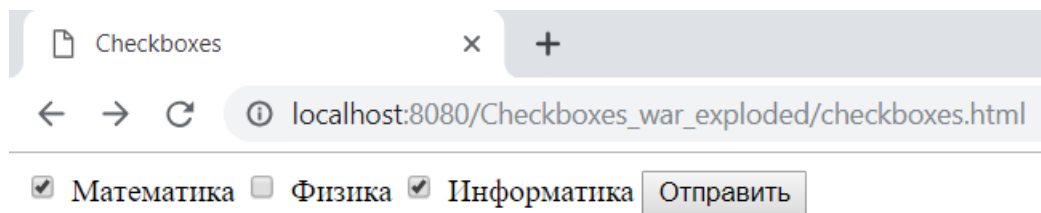


Рисунок 23 – Результат работы HTML-страницы

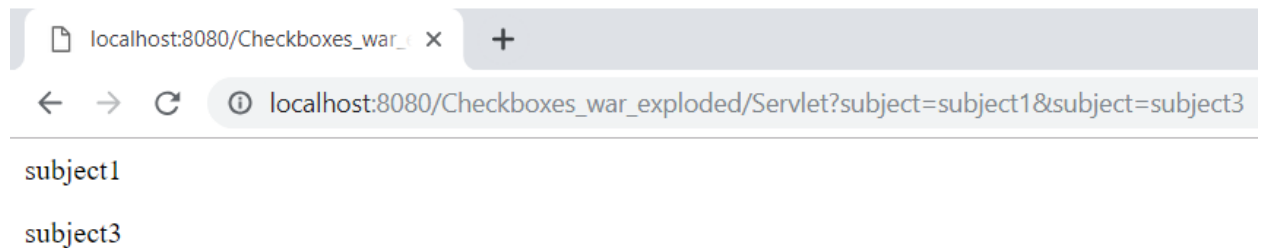


Рисунок 24 – Результат работы сервлета

Если нужно избавиться от имени артефакта в строке *URL*, то нужно проделать следующие действия (в данном случае артефакт имеет имя *Checkboxes_war_exploded*). Нажимаем *Edit Configurations...*, как показано на рисунке 25.

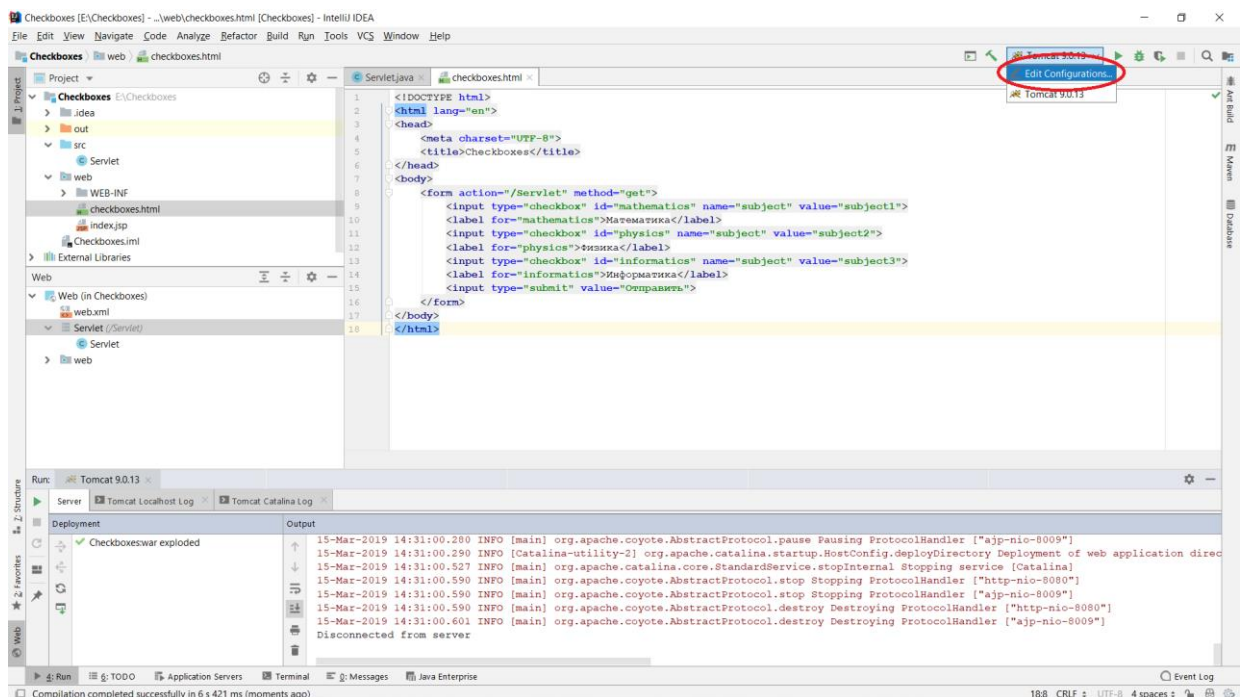


Рисунок 25 – Настройка конфигураций

Переходим во вкладку *Deployment*, где в поле *Application context* выбираем пустую строку. Нажимаем *OK* (рисунок 26).

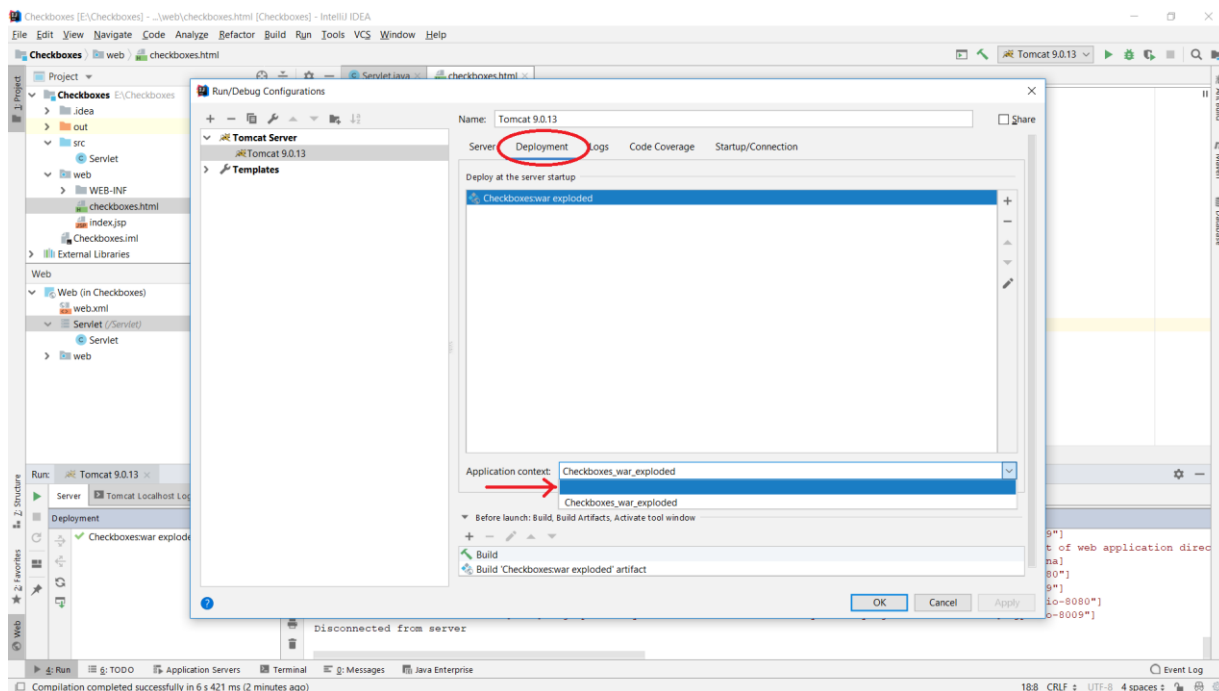


Рисунок 26 – Выбор контекста

После привычного запуска и ожидания получим результаты, которые представлены на рисунках 27 и 28 (обратите внимание на *URL*, имя артефакта уже не указывается).

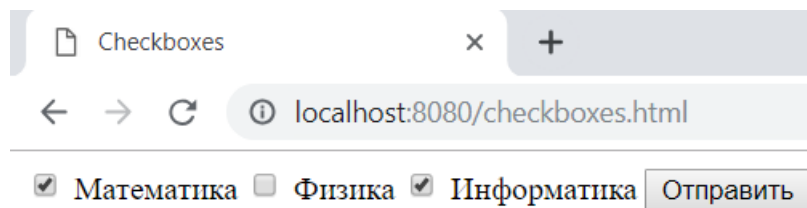


Рисунок 27 – Результат работы HTML-страницы

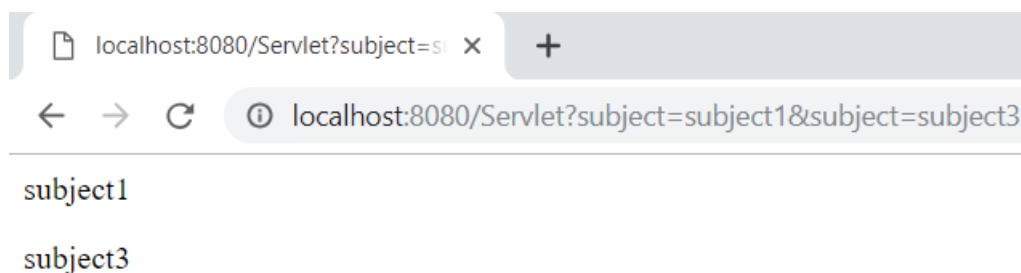


Рисунок 28 – Результат работы сервлета

Файл web.xml

В рассмотренном ранее примере мы не использовали файл *web.xml*. В самом сервлете мы помечали класс аннотацией `@WebServlet("/Servlet")`, которая содержала единственный параметр – строку. Эта строка соответствует *URL*, по которому будет доступен сервлет. Но можно обойтись и без этого. Откроем файл *web.xml*, он располагается по следующему пути: *web* → *WEB-INF* → *web.xml*). Содержимое данного файла показано на рисунке 29.

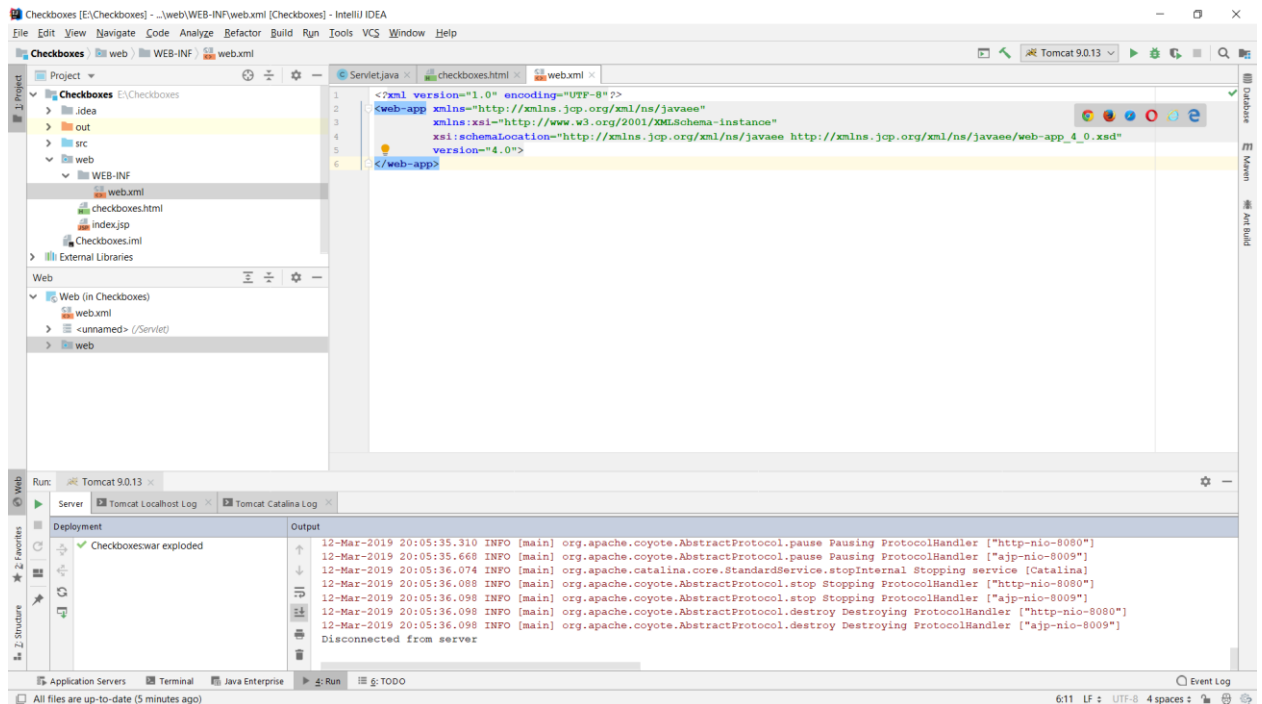


Рисунок 29 – Содержимое файла *web.xml*

Добавим в него строки, которые показаны на рисунке 30. Прежде всего, с помощью элемента `<servlet>` определяется сервлет. Элемент `<servlet-name>` задает имя сервлета, на которое будет проецироваться класс, указанный в элементе `<servlet-class>`. Имя может быть произвольным и может совпадать с названием класса. Затем в элементе `<servlet-mapping>` сервлет с именем *MyServlet* сопоставляется с путем `"/Servlet"`.

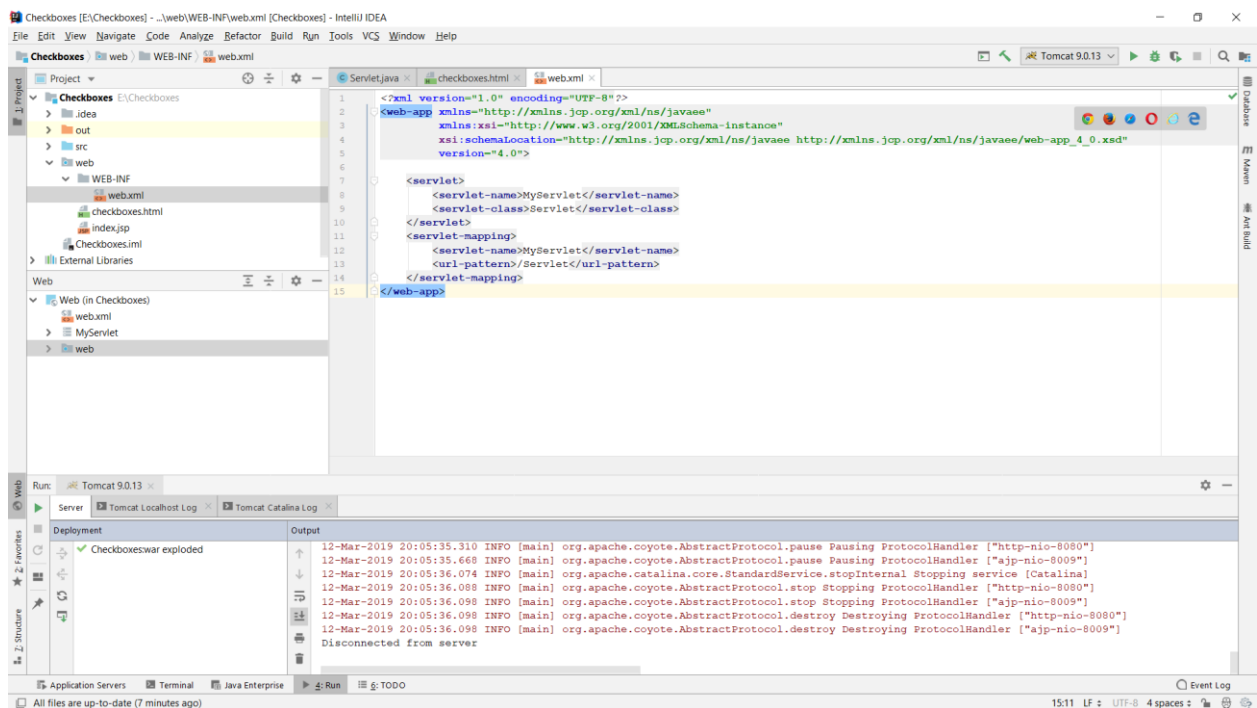


Рисунок 30 – Изменение содержимого файла web.xml

После запуска проекта мы получим аналогичные результаты, как если бы мы использовали аннотацию `@WebServlet("/Servlet")`. Данные результаты показаны на рисунке 31.

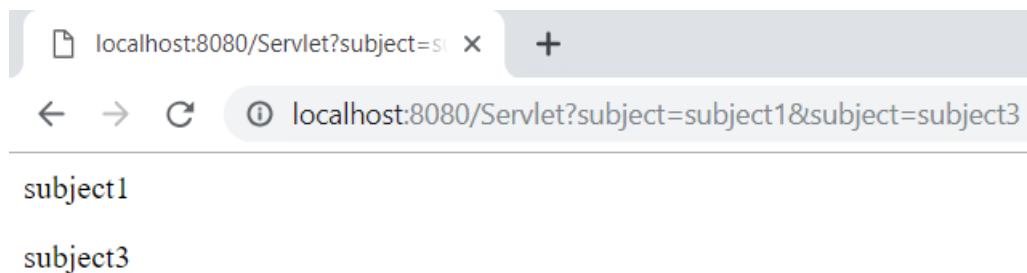


Рисунок 31 – Результат работы сервлета