# **Семинар №2 Название семинара**

1. **Инструментарий:**

[Презентация](https://docs.google.com/presentation/d/1Icg5YzRdfcNnIGMf28RakUeZUMJ4ktDK/edit?usp=sharing&ouid=117975875178229561167&rtpof=true&sd=true)

[Текст урока](https://docs.google.com/document/d/1Fj_PF43aUu7Jt0TnKxNtoFRrdRcswiBM/edit?usp=sharing&ouid=117975875178229561167&rtpof=true&sd=true)

1. **Цели семинара №2:**

* Получить базовые знания о фреймворке Spring и Spring Boot.
* Закрепить теоретические знания практическими заданиями.

По итогам семинара №2 слушатель должен **знать**:

● Что такое Spring Framework и для чего он используется.

● В чем преимущества Spring Boot и какие задачи он решает.

● Основные компоненты Spring и принципы их работы.

По итогам семинара №2 слушатель должен **уметь**:

* Создавать базовое приложение на Spring Boot.
* Понимать и использовать концепции Dependency Injection и Inversion of Control.
* Работать с базовыми компонентами Spring: Bean, ApplicationContext

1. **План Содержание:**

| **Этап урока** | **Тайминг, минуты** | **Формат** |
| --- | --- | --- |
| Коротко о Spring | 5 | Модерирует преподаватель |
| Создание пустого Spring приложения | 15 | Краткая демонстрация |
| Реализация базовой логики приложения | 15 | Краткая демонстрация |
| Задание из Блока 1 | 25 | Работа в команде |
| Обсуждение задания | 15 | Обсуждение с преподавателем |
| Задание из Блока 2 | 25 | Работа в команде |
| Обсуждение задания | 15 | Обсуждение с преподавателем |
| Заключение | 5 | Обзор пройденного материала |
| **Длительность:** | **120** |  |

### 

### **Блок 1.**

Тайминг:

Объяснение правил – 10 минут

Работа над заданием – 20 минут

Задание:

Создайте базовое приложение на Spring Boot, используя автоконфигурацию.

Пример решения:

Создание Spring Boot приложения с использованием автоконфигурации и Spring Boot Starter.

Часто встречающиеся ошибки:

Неэффективное использование автоконфигурации, непонимание работы Spring Boot Starter.

### **Блок 2.**

Тайминг:

Объяснение правил – 10 минут

Работа над заданием – 20 минут

Задание:

Создайте базовое приложение на Spring, используя Dependency Injection. В нем должен быть как минимум два бина, один из которых используется в другом. Названия этих бинов и предназначение может быть любым.

Пример решения:

**Создание проекта:** Переходим на Spring Initializr, выбираем нужные параметры и зависимости (например, "Spring Web" и "Spring Boot DevTools"), а затем создаем и скачиваем проект.

**Создание контроллера:** В проекте создаем класс "HelloController" со следующим содержимым:

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class HelloController {

@GetMapping("/")

public String hello() {

return "Hello, world!";

}

}

**Запуск приложения:** Запускаем приложение через среду разработки или Maven, а затем переходим в браузере по адресу "http://localhost:8080/" чтобы увидеть сообщение "Hello, world!".

Часто встречающиеся ошибки:

Неправильное использование аннотаций, неправильное определение Bean.

1. **Домашнее задание**

**Условие:**

Создать базовое веб-приложение с использованием Spring Boot, включающее в себя основные компоненты: контроллеры, сервисы и репозитории. Приложение должно быть простым, например, приложение для управления книжной библиотекой с операциями CRUD (создание, чтение, обновление и удаление) книг.

**Пример решения:**

**Создание проекта:** В Spring Initializr создаем новый проект с добавленной зависимостью "Spring Web".

**Создание бинов:** В проекте создаем два класса, HelloService и HelloController.

java

Copy code

import org.springframework.stereotype.Service;

@Service

public class HelloService {

public String getGreeting() {

return "Hello, world!";

}

}

Этот класс будет представлять собой сервис, который возвращает приветственное сообщение.

java

Copy code

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

import org.springframework.beans.factory.annotation.Autowired;

@RestController

public class HelloController {

private final HelloService helloService;

@Autowired

public HelloController(HelloService helloService) {

this.helloService = helloService;

}

@GetMapping("/")

public String hello() {

return helloService.getGreeting();

}

}

Этот класс будет контроллером, который использует HelloService для получения и отображения приветственного сообщения.

Здесь мы используем конструктор для внедрения зависимости HelloService в HelloController — это и есть Dependency Injection.

**Запуск приложения:** Запускаем приложение через среду разработки или Maven, а затем переходим в браузере по адресу "http://localhost:8080/" чтобы увидеть сообщение "Hello, world!".

Рекомендации для преподавателей по оценке задания:

- Убедитесь, что студент правильно создал и настроил проект Spring Boot.

- Проверьте, созданы ли необходимые классы и интерфейсы в соответствующих пакетах и имеют ли они правильные аннотации.

- Проверьте, правильно ли реализованы методы CRUD в классах сервиса и контроллера.

- Убедитесь, что приложение запускается и функционирует без ошибок.

- Проверьте, насколько хорошо студент понимает принципы работы Spring Boot, спросив его объяснить, как работает его код.

\*В задании не обязательно работать с базой данных, достаточно имитировать ее логику (хранить объекты в массиве).

\*Не обязательно реализовывать логику контроллера, достаточно просто класса, в котором вызываются методы ваших сервисов.