

**ПРАКТИЧЕСКАЯ ПРОГРАММА ОБУЧЕНИЯ**

# Spring Framework: IoC, JPA и MVC

## Понимание контейнера и разработка веб-приложения TaskFlow

**Java 21 • Spring Framework 7 • Spring Data JPA • Hibernate • PostgreSQL • Spring MVC • Thymeleaf**

| Общий объём | С преподавателем | Самостоятельно |
|-------------|------------------|----------------|
| 120 ак. ч.  | 40 ак. ч.        | 80 ак. ч.      |

## Проект после каждого модуля

TaskFlow: система задач с проектами, исполнителями, статусами, поиском и HTML-интерфейсом. Конфигурация Spring явная, данные хранятся в PostgreSQL, переходы состояния выполняются транзакционно.

### Обучение лично у основателя ТЕРАКОР

## Сергей Валерьевич Герасименко

Практикующий full-stack разработчик, технический директор и основатель ТЕРАКОР и ТЕРАКОР Digital, ведущий преподаватель и практикующий эксперт.

Лицензированный учебный центр • Живые онлайн-занятия

Редакция программы: 28 сентября 2026 года

## 01 / КОМУ ПОДОЙДЁТ КУРС

# Вход и результаты обучения

### Для кого

Java-разработчикам, которым важно понимать механизм Spring и подготовиться к Spring Boot.

### Что нужно знать до старта

Java Core, коллекции, исключения, JDBC и SQL; понимание интерфейсов и транзакций. Базовый JPA полезен, необходимое повторяется в модуле данных.

## После обучения вы сможете

- Настраивать контейнер и объяснять жизненный цикл бинов.
- Использовать Java Config, профили и внешнюю конфигурацию.
- Создавать репозитории и управлять транзакциями через Spring.
- Разрабатывать серверное MVC-приложение с формами и валидацией.
- Проверять контекст, веб-слой и транзакционное поведение.

## Как организована практика

Каждый модуль завершается проектным этапом с отдельным результатом и критериями приёмки. Этапы развивают единое приложение; при отдельном входе используется стартовый репозиторий.

Работа с преподавателем включает объяснение, практику, разбор ошибок и проверку результата. Самостоятельные часы включают завершение проектов, тесты и документацию.

**02 / УЧЕБНЫЙ ПЛАН**

# Распределение часов

| Модуль                                      | С преподавателем | Самостоятельно |
|---------------------------------------------|------------------|----------------|
| 1. IoC, DI и устройство Spring              | 4                | 8              |
| 2. Бины, области видимости и жизненный цикл | 4                | 8              |
| 3. Java Config, профили и внешние параметры | 6                | 12             |
| 4. Репозитории: JDBC, JPA и Spring Data     | 8                | 16             |
| 5. Транзакции и границы сервисного слоя     | 4                | 8              |
| 6. Spring MVC и серверные формы             | 8                | 16             |
| 7. AOP, наблюдаемость и проверки            | 4                | 8              |
| 8. Интеграция и защита TaskFlow             | 2                | 4              |
| ИТОГО                                       | 40               | 80             |

Все значения приведены в академических часах. Один академический час - 45 минут. Проекты и защита входят в указанный объём; дополнительные часы сверху не добавляются.

## 03 / МОДУЛЬ 01

# IoC, DI и устройство Spring

4 ак. ч. с преподавателем + 8 ак. ч. самостоятельно

## Содержание модуля

- Роль Spring и границы модулей экосистемы.
- Связанность кода; ручной DI и внедрение через конструктор.
- ApplicationContext, bean definition и граф зависимостей.
- Maven и согласованные версии Spring; Java 21.
- Каркас TaskFlow: domain, service, repository.

## Проект после модуля 1

## TaskFlow Core: сервис задач

Создать TaskService и репозиторий в памяти; сначала собрать зависимости вручную, затем через контейнер.

## Что сдаёт участник

Две конфигурации одного сценария создания задачи.

## Критерии приёмки

- Сервис зависит от интерфейса репозитория.
- Конструктор явно выражает обязательные зависимости.
- Обе сборки проходят один набор проверок.

## 03 / МОДУЛЬ 02

# Бины, области видимости и жизненный цикл

4 ак. ч. с преподавателем + 8 ак. ч. самостоятельно

## Содержание модуля

- Регистрация бинов и разрешение зависимостей.
- Singleton и prototype; request/session в контексте web.
- Инициализация и уничтожение; управление ресурсами.
- Циклические зависимости и диагностика ошибок контекста.
- XML-конфигурация: чтение и перенос простого примера в Java Config.

## Проект после модуля 2

## TaskFlow Context: управляемые компоненты

Настроить контейнер и компонент журналирования; показать lifecycle callbacks и исправить ошибочную зависимость.

### Что сдаёт участник

Конфигурация контекста и тесты получения компонентов.

### Критерии приёмки

- Контекст запускается без циклов.
- Область видимости выбрана и объяснена.
- Ресурсы закрываются при остановке контекста.

## 03 / МОДУЛЬ 03

# Java Config, профили и внешние параметры

6 ак. ч. с преподавателем + 12 ак. ч. самостоятельно

## Содержание модуля

- @Configuration, @Bean, @Component и component scan.
- @Qualifier и @Primary; неоднозначные зависимости.
- @PropertySource, Environment, @Value и безопасная конфигурация.
- @Profile: разные реализации для dev и test.
- SpEL: чтение выражений и простые сценарии; тестирование конфигурации.

## Проект после модуля 3

## TaskFlow Environments: конфигурация окружений

Создать dev/test-профили, переключать хранилище и уведомления без правок сервисов.

### Что сдаёт участник

Набор конфигураций и инструкции запуска профилей.

### Критерии приёмки

- Каждый профиль проходит smoke-тест контекста.
- Отсутствующий обязательный параметр выявляется при запуске.
- Секреты не хранятся в репозитории.

**03 / МОДУЛЬ 04**

# Репозитории: JDBC, JPA и Spring Data

**8 ак. ч. с преподавателем + 16 ак. ч. самостоятельно**

## Содержание модуля

- JdbcTemplate и перевод исключений доступа к данным.
- EntityManager и Hibernate; настройка инфраструктуры JPA в Spring.
- Сущности Task, Project, Assignee и связи.
- Spring Data JPA: методы запросов, JPQL, пагинация и DTO.
- N+1, fetch-план и SQL-диагностика; кэширование как отдельный компромисс.

**Проект после модуля 4**

## TaskFlow Persistence: постоянное хранение

Подключить PostgreSQL, создать репозитории, поиск по проекту и статусу; сравнить JDBC-и JPA-запрос для отчёта.

**Что сдаёт участник**

Схема данных, репозитории и интеграционные тесты.

**Критерии приёмки**

- Поиск и пагинация возвращают ожидаемые результаты.
- Список задач не порождает запрос на каждого исполнителя.
- Ограничения схемы не позволяют создать задачу с несуществующим проектом.

**03 / МОДУЛЬ 05**

# Транзакции и границы сервисного слоя

**4 ак. ч. с преподавателем + 8 ак. ч. самостоятельно**

## Содержание модуля

- ACID и граница бизнес-операции.
- @Transactional: propagation, isolation, readOnly и rollbackFor.
- Прокси и self-invocation: почему аннотация может не сработать.
- TransactionTemplate для программного сценария.
- Проверка rollback на PostgreSQL; конфликт изменений.

**Проект после модуля 5**

## TaskFlow Workflow: атомарная смена статуса

Изменять статус задачи и записывать историю одной транзакцией. Воспроизвести ошибку сохранения истории.

**Что сдаёт участник**

Транзакционный сервис и тесты отката.

**Критерии приёмки**

- Ошибка истории откатывает изменение задачи.
- Вызов через Spring проху и прямой внутренний вызов различаются в объяснении.
- Тест проверяет состояние БД после завершения транзакции.

**03 / МОДУЛЬ 06**

# Spring MVC и серверные формы

**8 ак. ч. с преподавателем + 16 ак. ч. самостоятельно**

## Содержание модуля

- DispatcherServlet, controller, model и view resolver.
- @RequestMapping, @PathVariable, @RequestParam, @ModelAttribute.
- Thymeleaf: список, карточка, создание и редактирование.
- Jakarta Validation, BindingResult, сообщения об ошибках.
- Post/Redirect/Get, поиск, пагинация и обработка исключений.

**Проект после модуля 6**

## TaskFlow Web: рабочий интерфейс

Создать HTML-интерфейс задач и проектов с формами, фильтрами и корректными сообщениями валидации.

**Что сдаёт участник**

MVC-приложение и проверки контроллеров.

**Критерии приёмки**

- Некорректная форма показывает ошибки и сохраняет введенные данные.
- Обновление страницы после POST не создаёт дубль.
- Пустой список и отсутствующая задача обработаны явно.

---

**03 / МОДУЛЬ 07**

# АОР, наблюдаемость и проверки

**4 ак. ч. с преподавателем + 8 ак. ч. самостоятельно**

## Содержание модуля

- Proxy-based AOP: join point, pointcut, advice.
- @Around для времени выполнения и диагностики ошибок.
- Ограничения прокси; взаимосвязь AOP и @Transactional.
- Spring TestContext и MockMvc; слои проверки.
- Логи без чувствительных данных; избегаем дублирования сквозной логики.

### Проект после модуля 7

## TaskFlow Diagnostics: измерение операций

Добавить аспект времени выполнения сервисов и тесты MVC; проверить успешную операцию и исключение.

### Что сдаёт участник

Диагностический аспект и набор автоматических проверок.

### Критерии приёмки

- Аспект не меняет результат и тип исключения.
- Измеряется нужный слой, а не каждый метод без разбора.
- Логи позволяют найти неуспешную операцию.

---

**03 / МОДУЛЬ 08**

# Интеграция и защита TaskFlow

**2 ак. ч. с преподавателем + 4 ак. ч. самостоятельно**

## Содержание модуля

- Проверка полного пользовательского пути.
- Конфигурация PostgreSQL и воспроизводимый запуск.
- Разбор границ транзакций и зависимостей.
- Демонстрация проекта и адресная обратная связь.

### Проект после модуля 8

## TaskFlow Release: итоговый релиз

Собрать результаты модулей, показать создание проекта, назначение задачи, смену статуса и историю.

### Что сдаёт участник

Версия проекта с README, схемой компонентов и тестами.

### Критерии приёмки

- Проект запускается по инструкции на чистом стенде.
- Показаны MVC-валидация и транзакционный rollback.
- Автор объясняет DI, проху и отличие JPA от Spring Data.

## 04 / ИТОГОВЫЙ РЕЗУЛЬТАТ

# Проект для портфолио

TaskFlow: система задач с проектами, исполнителями, статусами, поиском и HTML-интерфейсом. Конфигурация Spring явная, данные хранятся в PostgreSQL, переходы состояния выполняются транзакционно.

## Комплект сдачи

- Репозиторий с исходным кодом и понятной структурой.
- README: требования к окружению, команды сборки и запуска, пример использования.
- Конфигурация и данные для воспроизведения проверок без секретов.
- Результаты тестов или проверочные сценарии для каждого проектного этапа.
- Краткое описание архитектуры, ограничений и принятых решений.

## Порядок защиты

Участник запускает приложение, показывает основной сценарий и обработку ошибки, объясняет ключевые решения. Проверка опирается на критерии каждого модуля. Невыполненный критерий фиксируется как конкретная доработка.

### Условие завершения

Все обязательные проектные этапы приняты; итоговый сценарий воспроизводится по README. Итоговый проект объединяет ранее выполненные этапы и не является дополнительной работой сверх программы.

### Результат обучения

Работающий учебный проект и подтверждённые практикой навыки. Прохождение отдельного курса не означает автоматического присвоения грейда или гарантии трудоустройства.