Skillbox

Bean: Жизненный цикл и scope

Кирилл Кошаев

Java-разработчик в «Газпром информ»

Цель и задачи урока

Цель: выяснить, что такое bean в Spring

- **Задача №1:** рассмотреть жизненный цикл bean.
- **Задача №2:** выяснить, что такое bean scope.

Bean

JavaBean:

A. **Должен** имплементировать интерфейс Serializable.
B. Поля должны иметь **только** private доступ.
C. **Должен** иметь пустой конструктор.
D. Для полей **должны** быть определены get и set методы.

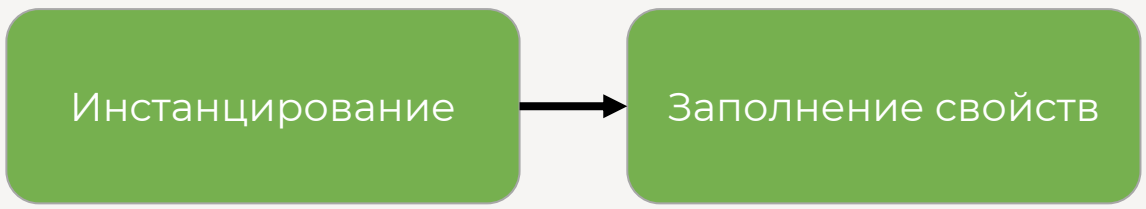
Spring bean:

A. **Может** имплементировать интерфейс Serializable.
B. Поля могут иметь **любой** уровень доступа.
C. **Может** иметь пустой конструктор.
D. Для полей **не обязательно** определять get и set методы.

Skillbox

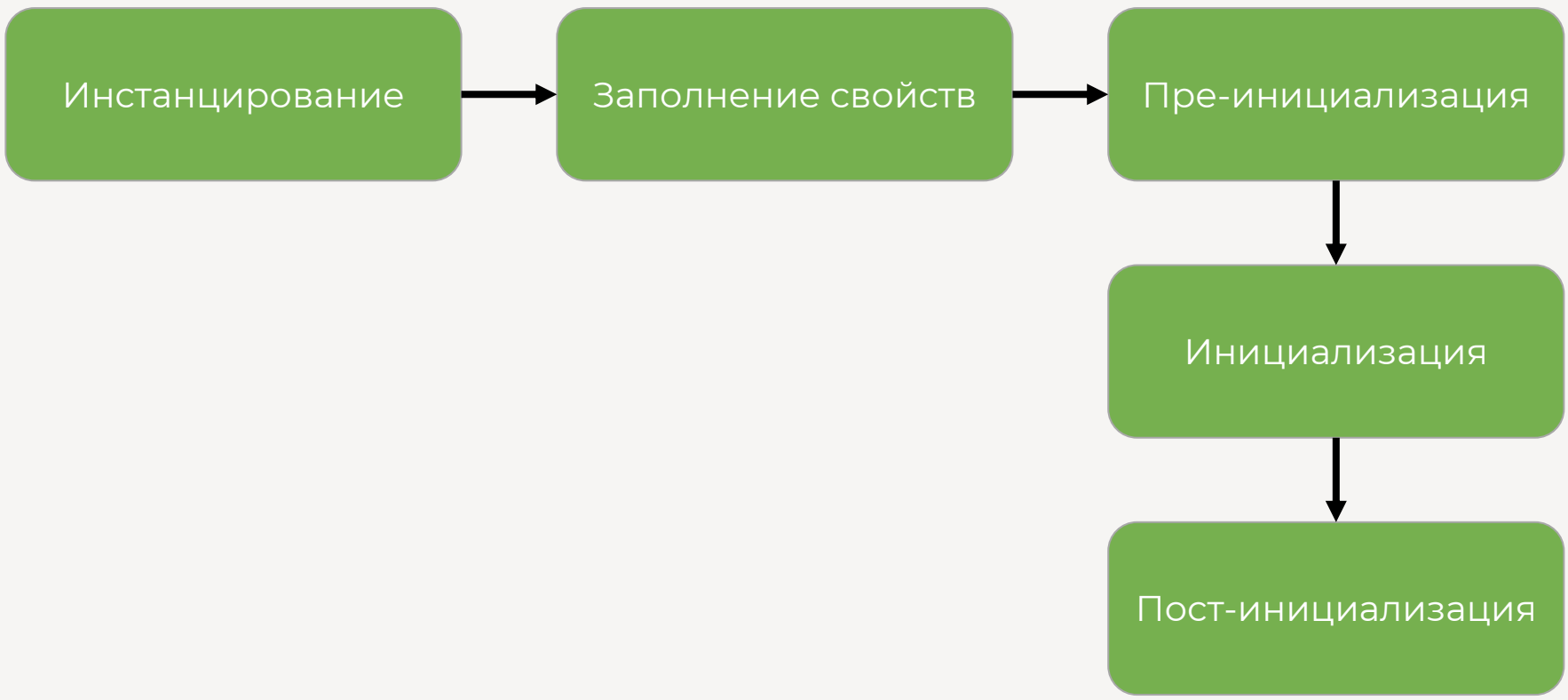
Жизненный цикл bean

Инстанцирование объекта — технически, начало жизни бина,
во время которого происходит создание экземпляра объекта.
Заполнение свойств — установка значений и внедрение зависимостей



Жизненный цикл bean

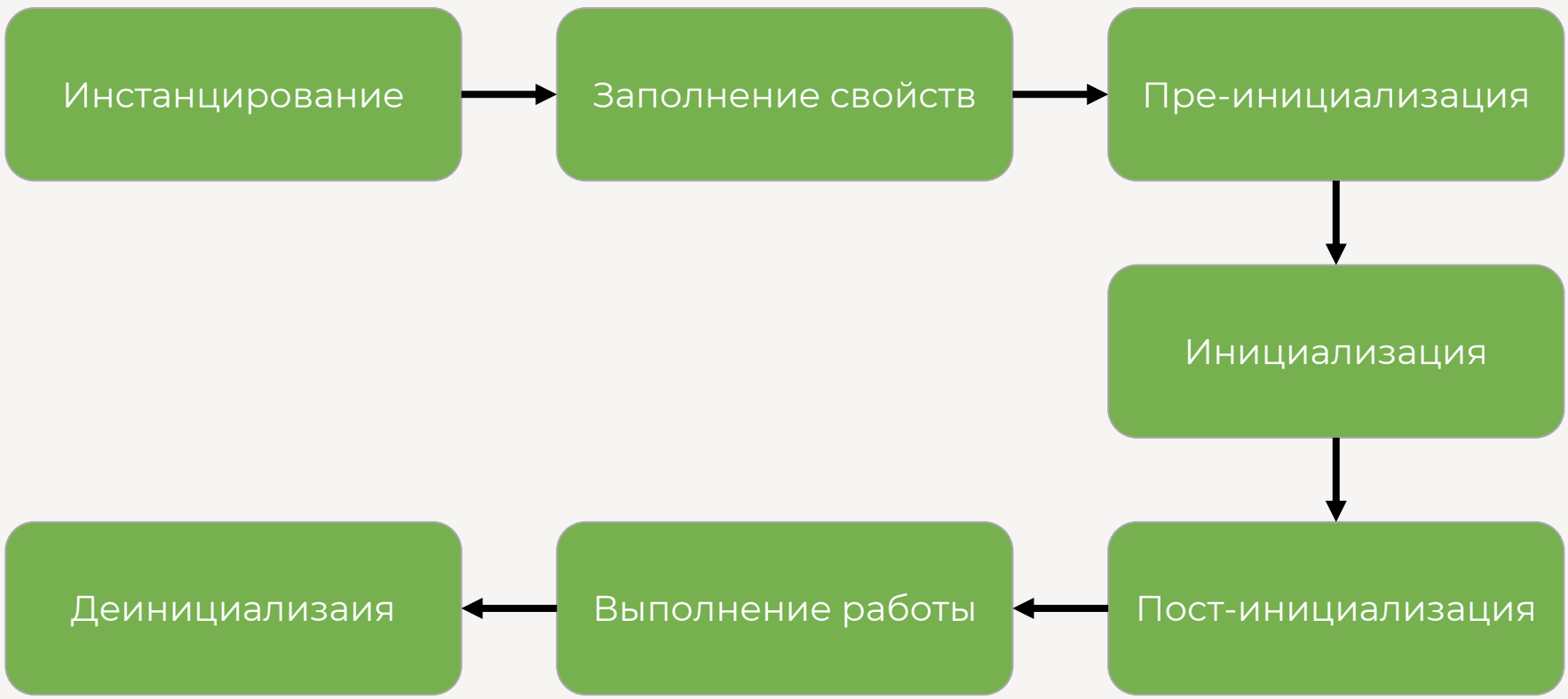
- Пре-инициализация — вызывается код, выполняемый перед основной инициализацией.
- Инициализация бина — выполняется основная работа по инициализации.
- Пост-инициализация выполняется перед тем, как Bean будет задействован в работе.



Skillbox

Жизненный цикл bean

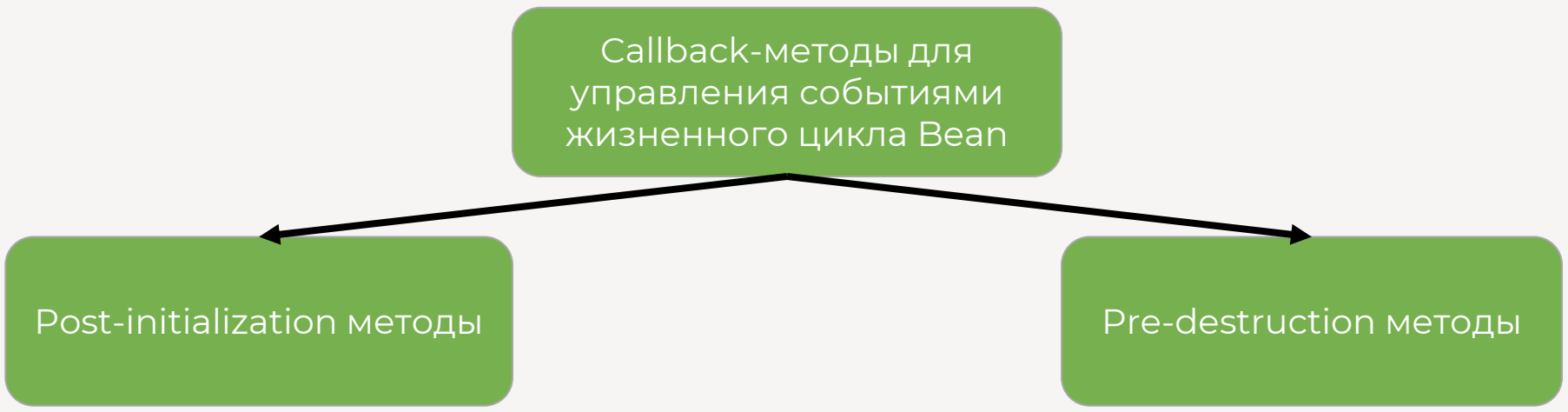
Деинициализация — этап жизненного цикла бина, в ходе которого происходит уничтожение экземпляра бина и освобождение ресурсов



Skillbox

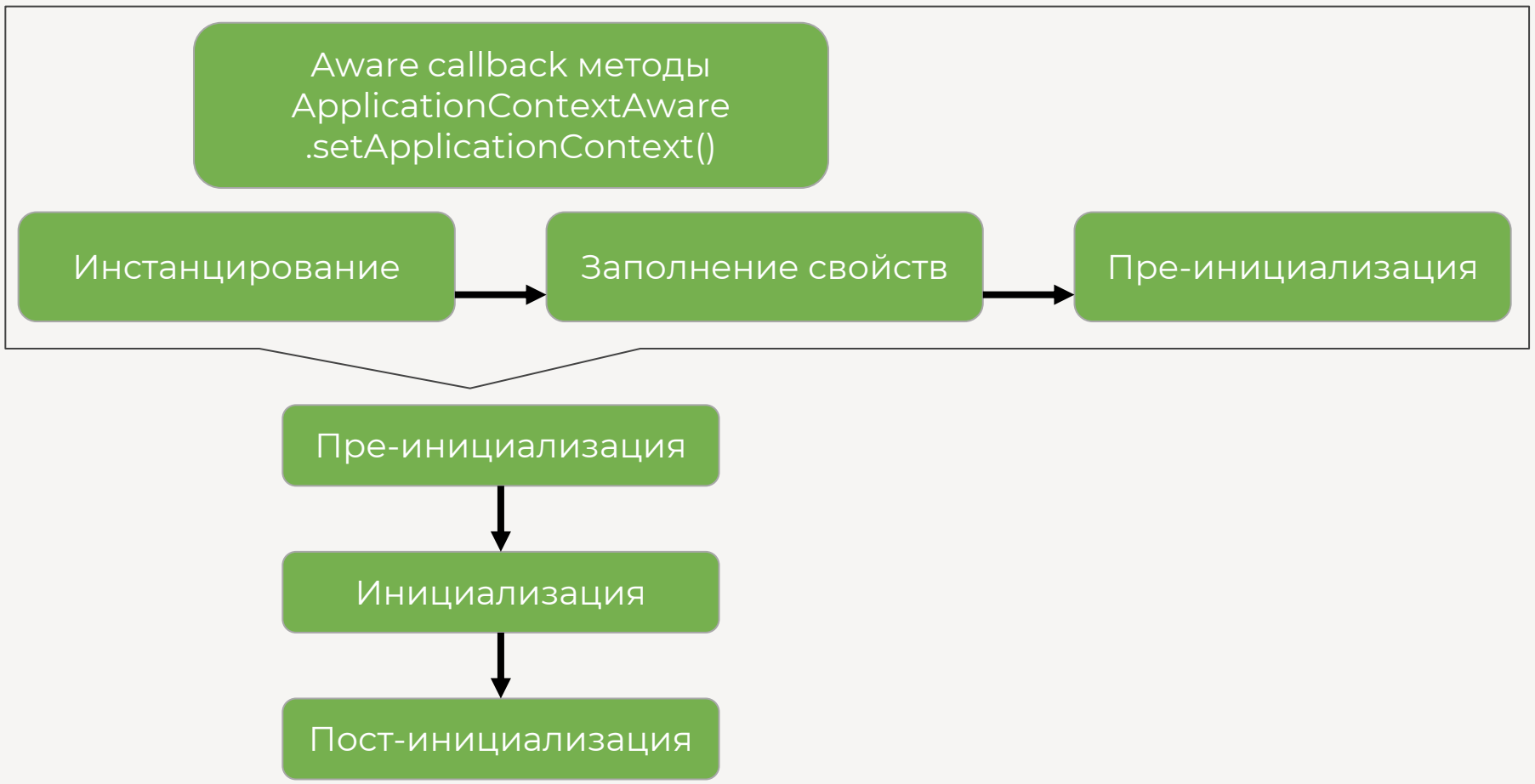
Управление жизненным циклом bean

BeanFactory предоставляет различные callback-методы, которые широко могут быть разделены на две группы: Post-Initialization и Pre-destruction callback-методы



Aware интерфейсы

Интерфейсы — осведомители, которые позволяют бинам оповещать контейнер о том, что им для работы требуются некоторые зависимости, которые не относятся к бинам самого приложения



Skillbox

Aware интерфейсы

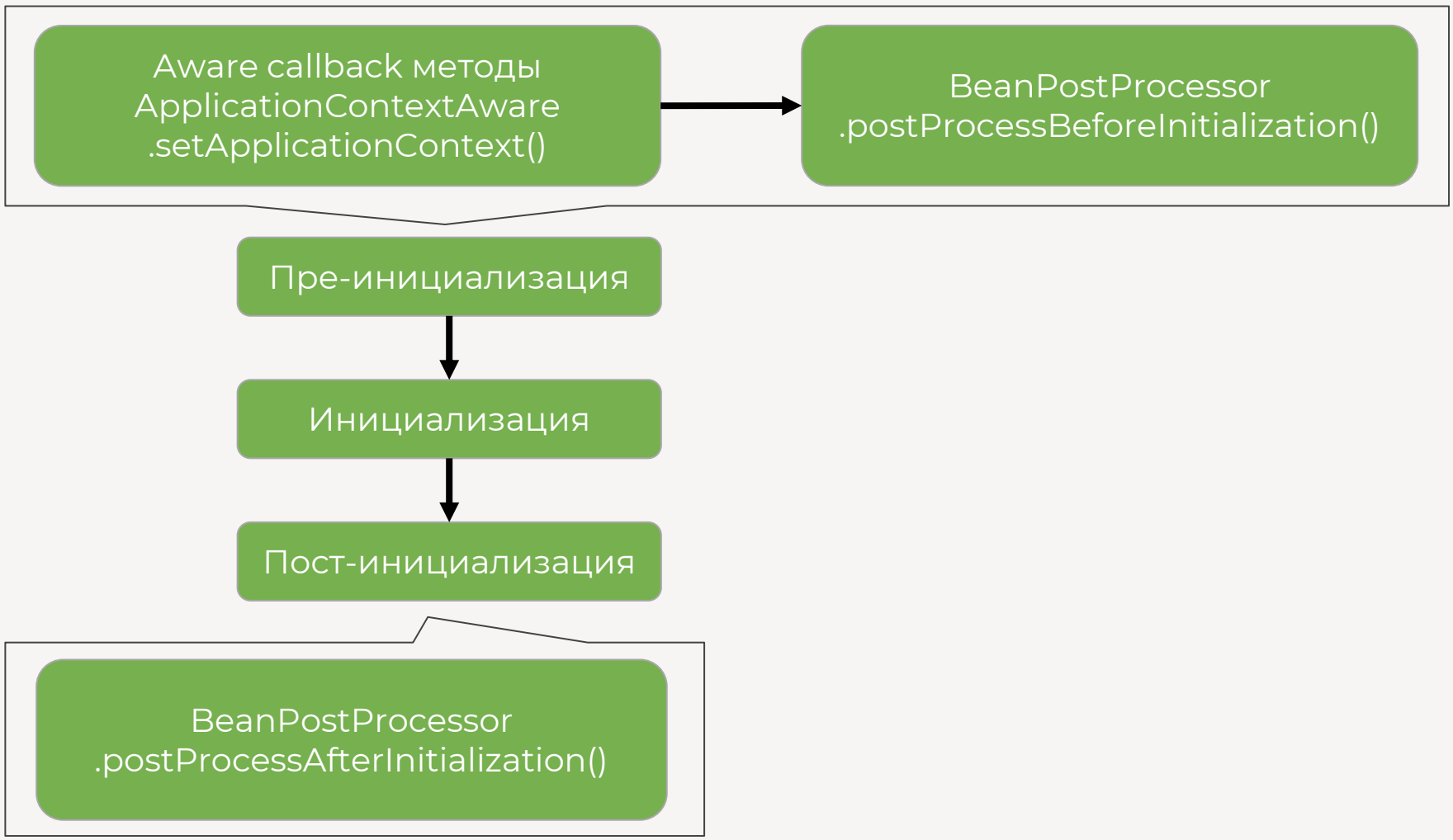
Интерфейсы-осведомители, которые позволяют бинам оповещать контейнер о том, что им для работы требуются некоторые зависимости, которые не относятся к бинам самого приложения

```
public class DemoBean implements ApplicationContextAware
{
    @Override
    public void setApplicationContext(ApplicationContext arg0)
            throws BeansException {
        // TODO Auto-generated method stub
    }
}
```

Skillbox

BeanPostProcessor

Интерфейс, который определяет два метода:
postProcessBeforeInitialization(), postProcessAfterInitialization()



BeanPostProcessor

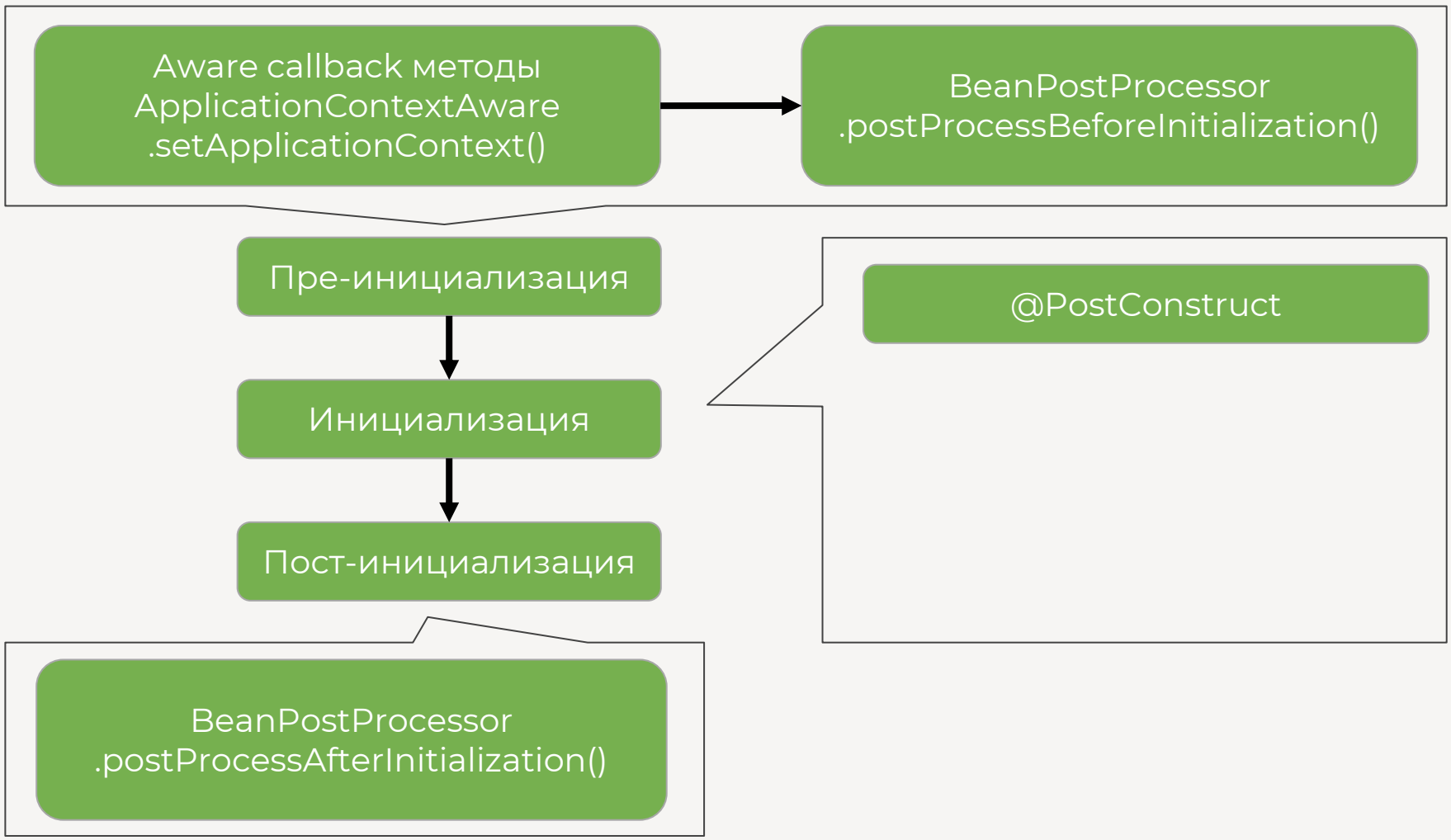
Интерфейс определяет два метода:
postProcessBeforeInitialization(), postProcessAfterInitialization()

```
import org.springframework.beans.BeansException;
import org.springframework.beans.factory.config.BeanPostProcessor;
public class CustomBeanPostProcessor implements BeanPostProcessor
{
    @Override
    public Object postProcessBeforeInitialization(Object bean, String beanName) throws
BeansException
    {
        System.out.println("Called postProcessBeforeInitialization() for :" + beanName);
        return bean;
    }
    @Override
    public Object postProcessAfterInitialization(Object bean, String beanName) throws
BeansException
    {
        System.out.println("Called postProcessAfterInitialization() for :" + beanName);
        return bean;
    }
}
<beans>
     <bean id="customBeanPostProcessor"
                class="com.howtodoinjava.demo.processors.CustomBeanPostProcessor" />
</beans>
```

Skillbox

@PostConstruct

@PostConstruct метод будет вызван после того, как бин был создан при помощи конструктора по умолчанию



@PostConstruct

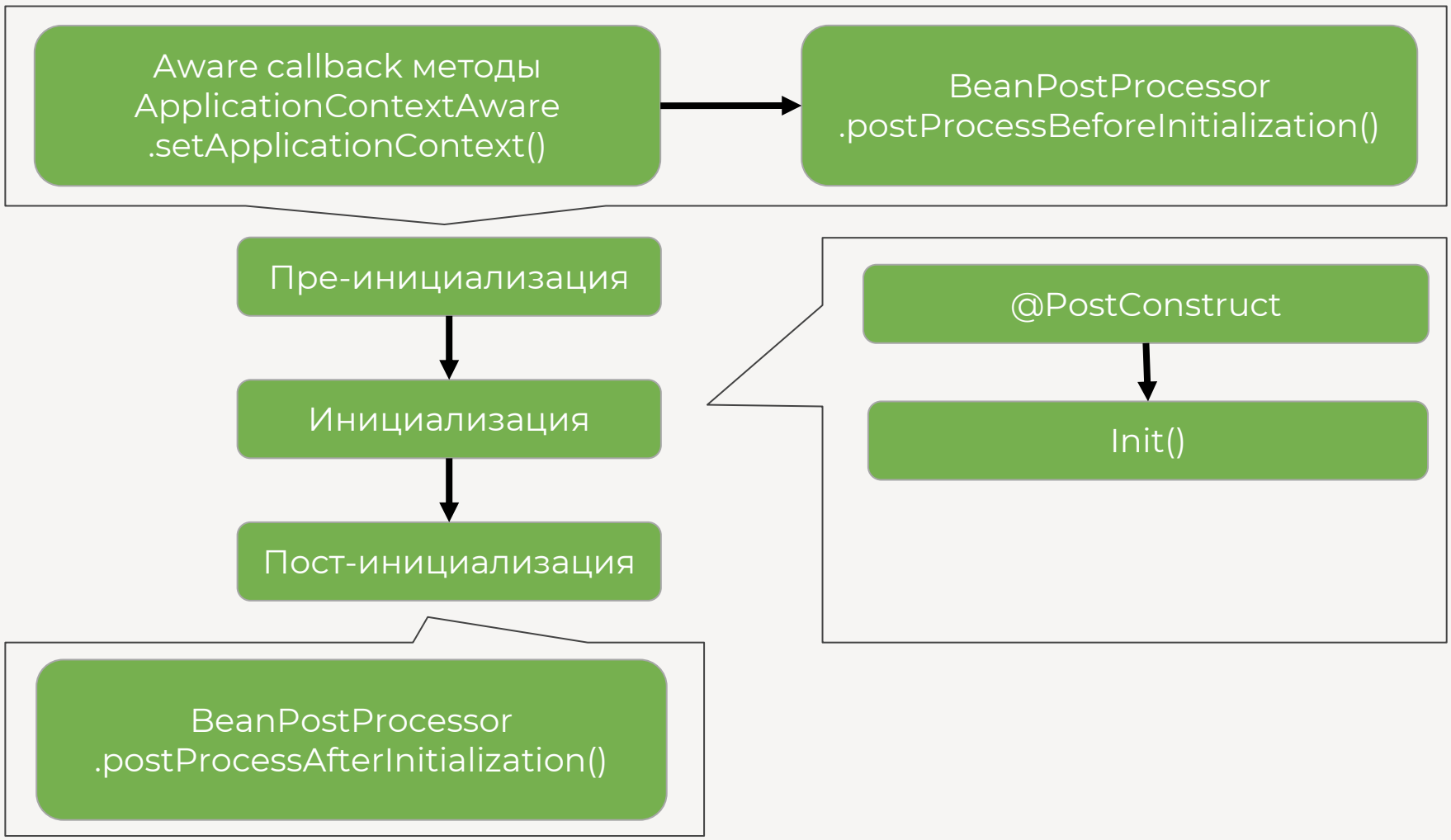
@PostConstruct метод будет вызван после того, как бин был создан при помощи конструктора по умолчанию

```
import javax.annotation.PostConstruct;
import javax.annotation.PreDestroy;

public class DemoBean
{

    @PostConstruct
    public void customInit()
    {
        System.out.println("Method customInit() invoked...");
    }

}
```

Skillbox

Init методы

Можно определять локальные для каждого бина, а также общие для всех, глобальные, init и методы в конфигурационном файле



Init методы

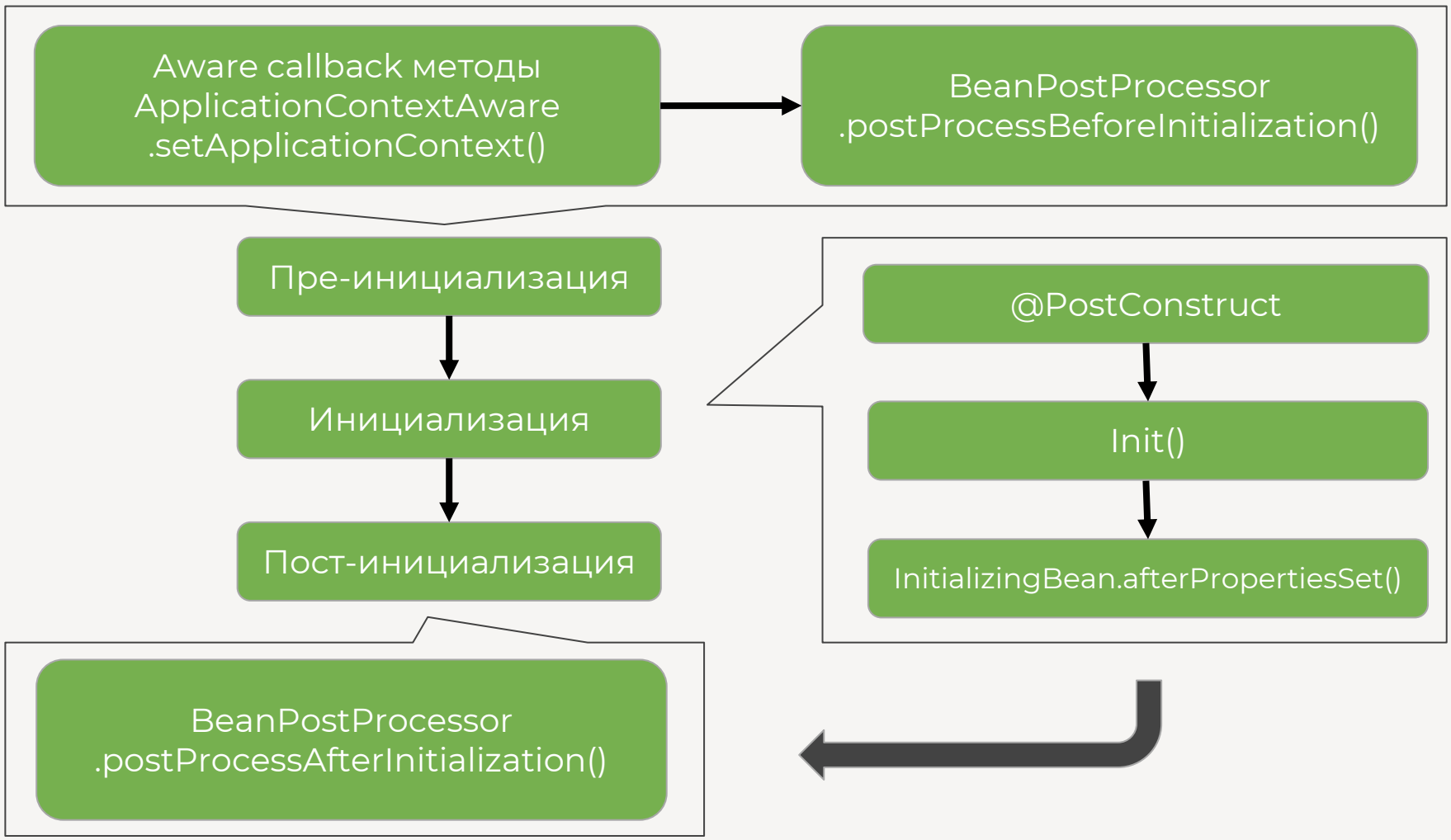
Можно определять локальные для каждого бина, а также общие для всех глобальные init и методы в конфигурационном файле

```
<beans default-init-method="customInit">
    <bean id="demoBean" class="com.howtodoinjava.task.DemoBean"
                    init-method="beanCustomInit">
    </bean>
</beans>

public class DemoBean
{
    public void customInit()
    {
        System.out.println("Method customInit() invoked...");
    }
    public void beanCustomInit()
    {
        System.out.println("Method beanCustomInit() invoked...");
    }
}
```

Skillbox

InitializingBean

Интерфейс InitializingBean позволяет бину выполнять некоторую работу на этапе инициализации после того, как все необходимое свойства и зависимости были установлены



InitializingBean

Интерфейс InitializingBean позволяет бину выполнять некоторую работу на этапе инициализации после того, как все необходимые свойства и зависимости были установлены

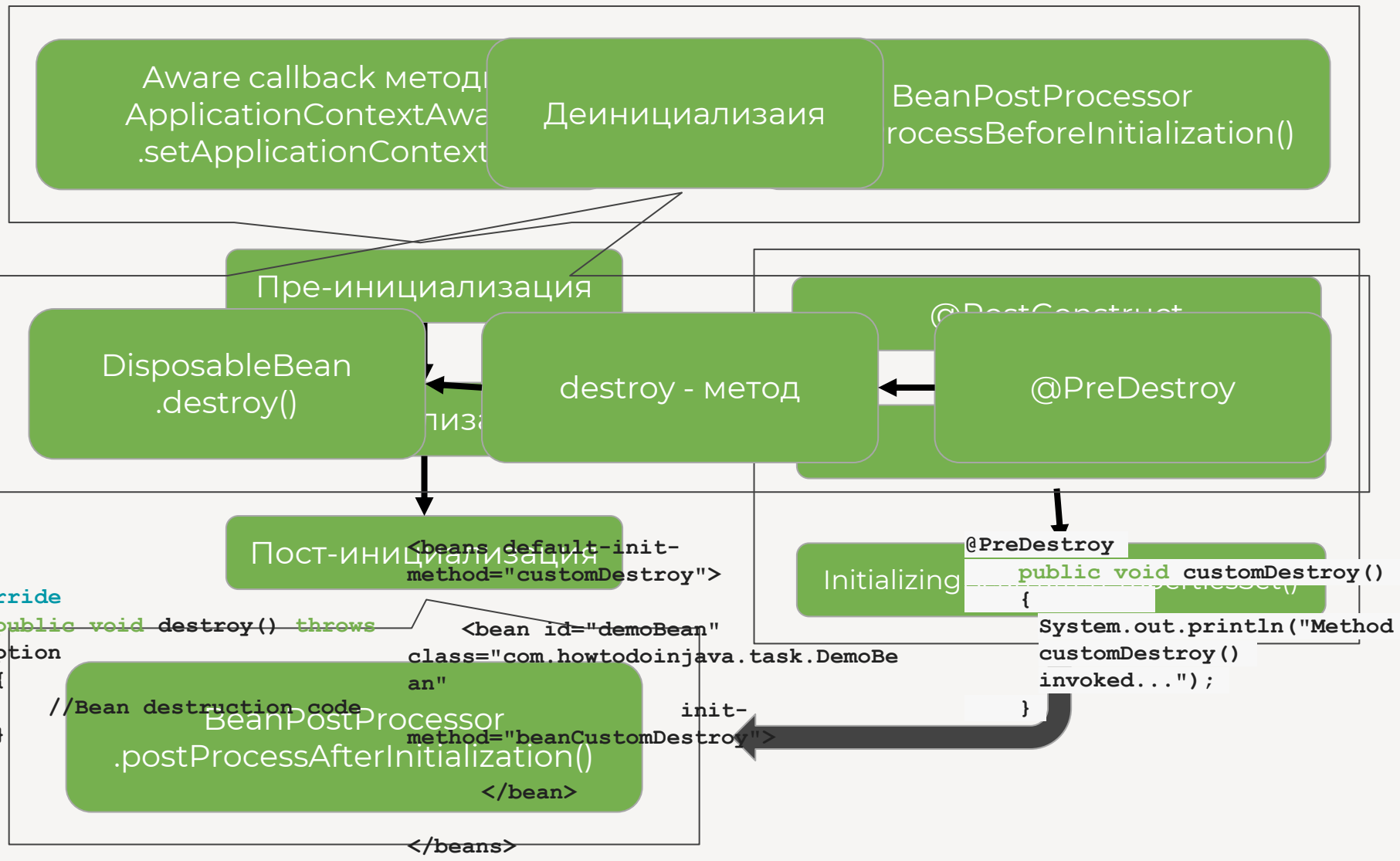
```
import org.springframework.beans.factory.InitializingBean;

public class DemoBean implements InitializingBean
{
    //Other bean attributes and methods

    @Override
    public void afterPropertiesSet() throws Exception
    {
        //Bean initialization code
    }

}
```

Skillbox

Деинициализация

Когда бин выполнил свою работу и больше не нужен, или контейнер завершает свою работу, начинается этап деинициализации, в ходе которого происходит уничтожение экземпляра бина и освобождение ресурсов

Деинициализация



Skillbox

@Scope

Помимо жизненного цикла у бинов есть ещё такая характеристика, как scope — область видимости Spring bean

- @Scope(“singleton”) — один экземпляр на один контейнер;
- @Scope(“prototype”) — новый экземпляр при каждом вызове в коде;
- @Scope(“request”) — один экземпляр на каждый http-запрос;
- @Scope(“session”) — один экземпляр на http-сессию;
- @Scope(“application”) — один экземпляр на каждый контекст приложения;
- @Scope(“websocket”) — один экземпляр на websocket-соединение.

@Scope

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd">

    <bean id="personSingleton" class="org.baeldung.scopes.Person"
scope="singleton"/>

</beans>
```

Skillbox

Спасибо за внимание!