BRING FRAMEWORK

Brings the objects your application needs

Bobocode

Initial requirements. Bring project

The initial and the most important part of this project is **Bring IoC – an inversion of control and dependency injection container.** It allows you to **declare what objects you need** for the application and then **brings all those objects**. The task of object creation may seem trivial. But when it comes to real applications that require **hundreds or thousands of objects that are connected,** it becomes much tricker. Some of those objects require complex initialization, and some of them require additional post-construction logic. So the Bring IoC focuses on the infrastructural part of the application and brings you all objects that are ready to use.

Key parts

Two major parts of Bring that users will work with are:

1. configuration
2. context

Configuration

A configuration provides the instructions for the container. It tells which objects should be created, and what values should be injected into the fields. So Bring IoC should provide some tools for the configuration. It could be XML files, JSON files, property text files, Java, Groovy, or Kotlin classes, annotations, etc. Think about the following question:

What is the most convenient way to tell the framework what objects need to be created?

Context

When everything is configured and created, it should be stored somewhere. In other words, all the **objects that were created by the container should be stored in some context** (register of objects). This context should provide and ability to fetch the objects from it. In other words, if I specified that I need an instance of class A for my application, then when everything is created, I should be able to tell the container: "give me that instance of class A".

Container internals

As long as it provides an easy way to declare what objects you need, and then brings those objects, people don't care much about other stuff. E.g. how the container works under the hood. Please note that there are **multiple ways to do the same thing** so there will be **a lot of edge cases**. Please take a look at some of them:

- an empty constructor
- multiple constructors
- a constructor for required final fields
- a private constructor
- multiple implementations of the same interface
- multiple instances of the same class
- circular dependencies between classes

Ideally, **your implementation should cover all of them, but if not, it should still be handled in a proper way.** E.g. it should be documented, and in the code, it should fail gracefully with proper exceptions.

The checklist

- Bring framework should provide a convenient tool for configuration (so a user can easily tell what objects he/she wants)
- given the configuration, it should be able to create all required objects and inject dependencies handling various edge cases
- if something went wrong, and the context cannot be created, it should fail gracefully and provide a detailed description of the problem and/or instructions on how to fix it
- the API should provide an easy way to fetch created objects from the context, and handle cases when the required object is not found
- the implementation should be covered with tests and useful documentation (Javadoc)
- the repository should have a nice and easy to understand README file with instructions for users

The goal

You are going to build a Bring IoC from scratch. Working on this project, you'll need to dive into the problems related to inversion of control and dependency injection. Along the way, you'll make design decisions and work on low-level technical issues. As a team, you'll need to figure a way how to get this done. Remember, that **the main purpose of this assignment is to learn**. And you have a variety of possibilities for learning.

Learning Objectives

- better understand how Spring IoC works
- learn and use Reflection API
- apply design patterns
- learn how to write better tests
- practice clean code principles
- learn how to create a useful documentation
- work within the team
- participate in code reviews