



CS544 EA
Spring

Startup: Lazy and Eager

Eager

- By default beans are created eagerly
 - Right away when the context is created

With Java Config and Annotations:

```
package cs544.spring27.startup.eager;

...

@Configuration
@ComponentScan("cs544.spring27.startup.eager")
public class Config {
}
```

With XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans" ...
  <bean id="customerService"
        class="cs544.spring27.startup.eager.CustomerService"
        init-method="start" destroy-method="stop"/>
</beans>
```

```
package cs544.spring27.startup.eager;

...

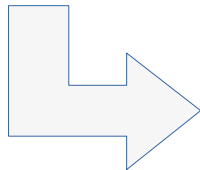
@Service
public class CustomerService {
    public CustomerService() {
        System.out.println("constructor");
    }
    @PostConstruct
    public void start() {
        System.out.println("start");
    }
    public void hello() {
        System.out.println("hello");
    }
    @PreDestroy
    public void stop() {
        System.out.println("stop");
    }
}
```

Demonstration of Eager

```
package cs544.spring27.startup.eager;

import org.springframework.context.ConfigurableApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;

public class App {
    public static void main(String[] args) {
        System.out.println("Before context created");
        ConfigurableApplicationContext context;
        context = new AnnotationConfigApplicationContext(Config.class);
        System.out.println("After context, before getBean");
        CustomerService cs = context.getBean("customerService", CustomerService.class);
        System.out.println("After getBean, before bean use");
        cs.hello();
        System.out.println("After bean use, before close");
        context.close();
        System.out.println("After context close");
    }
}
```



Before context created
constructor
start
After context, before getBean
After getBean, before bean use
hello
After bean use, before close
stop
After context close

Lazy

- You can tell beans to be 'lazy'
 - Not created until needed for the first time (for injection or getBean)

With Annotations:

```
package cs544.spring28.startup.lazy;

...

@Service
@Lazy
public class CustomerService {
    ... // same content as eager
}
```

With XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans" ...
    <bean id="customerService" lazy-init="true"
        class="cs544.spring27.startup.eager.CustomerService"
        init-method="start" destroy-method="stop"/>
</beans>
```

With Java Config:

```
package cs544.spring28.startup.lazy;

...

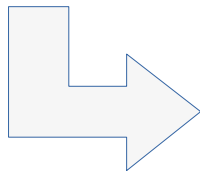
@Configuration
public class Config {
    @Bean
    @Lazy
    public CustomerService customerService() {
        return new CustomerService();
    }
}
```

Demonstration of Lazy

```
package cs544.spring27.startup.eager;

import org.springframework.context.ConfigurableApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;

public class App {
    public static void main(String[] args) {
        System.out.println("Before context created");
        ConfigurableApplicationContext context;
        context = new AnnotationConfigApplicationContext(Config.class);
        System.out.println("After context, before getBean");
        CustomerService cs = context.getBean("customerService", CustomerService.class);
        System.out.println("After getBean, before bean use");
        cs.hello();
        System.out.println("After bean use, before close");
        context.close();
        System.out.println("After context close");
    }
}
```



```
Before context created
After context, before getBean
constructor
start
After getBean, before bean use
hello
After bean use, before close
stop
After context close
```

Prototype = Lazy

- Prototype beans are always lazily instantiated
 - Not created until needed (may need to create many)

```
package cs544.spring28.startup.lazy;  
  
...  
  
@Service  
@Scope("prototype")  
public class CustomerService {  
    ...  
}
```

```
package cs544.spring27.startup.eager;  
  
import org.springframework.context.ConfigurableApplicationContext;  
import org.springframework.context.annotation.AnnotationConfigApplicationContext;  
  
public class App {  
    public static void main(String[] args) {  
        System.out.println("Before context created");  
        ConfigurableApplicationContext context;  
        context = new AnnotationConfigApplicationContext(Config  
        System.out.println("After context, before getBean");  
        CustomerService cs = context.getBean("customerService",  
        System.out.println("After getBean, before bean use");  
        cs.hello();  
        System.out.println("After bean use, before close");  
        context.close();  
        System.out.println("After context close");  
    }  
}
```

Almost same as
@Lazy

Before context created
After context, before getBean
constructor
start
After getBean, before bean use
hello
After bean use, before close
After context close

No destroy / stop