Capgemini

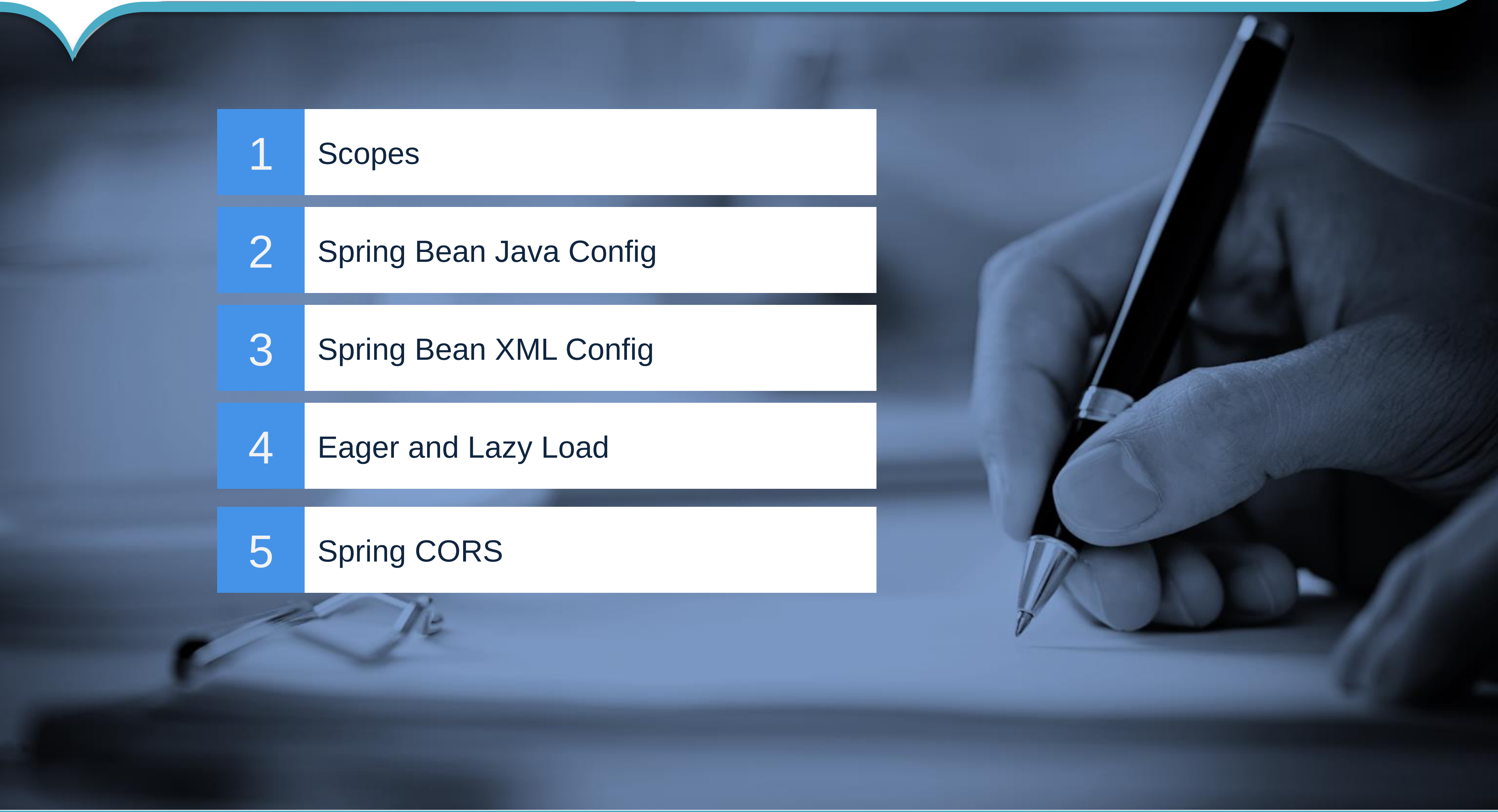
CONSULTING.TECHNOLOGY.OUTSOURCING

Spring5

2018

People matter, results count.

Agenda



1 Scopes

2 Spring Bean Java Config

3 Spring Bean XML Config

4 Eager and Lazy Load

5 Spring CORS

Scopes

Scopes

- Singleton
- Prototype
- Request
- Session
- Application
- WebScocket
- Custom Thread

Scopes

SCOPE	DESCRIPTION
singleton(default)	Single bean object instance per spring IoC container
prototype	Opposite to singleton, it produces a new instance each and every time a bean is requested.
request	A single instance will be created and available during complete lifecycle of an HTTP request. Only valid in web-aware Spring ApplicationContext.
session	A single instance will be created and available during complete lifecycle of an HTTP Session. Only valid in web-aware Spring ApplicationContext.
application	A single instance will be created and available during complete lifecycle of ServletContext. Only valid in web-aware Spring ApplicationContext.
websocket	A single instance will be created and available during complete lifecycle of WebSocket. Only valid in web-aware Spring ApplicationContext.

Custom Thread Scope

```
<bean
class="org.springframework.beans.factory.config.CustomScopeConfigurer">
    <property name="scopes">
        <map>
            <entry key="thread">
                <bean
class="org.springframework.context.support.SimpleThreadScope"/>
            </entry>
        </map>
    </property>
</bean>
```

```
@Component
@Scope("thread")
public class BeanClass {
}
```

```
<bean id="beanId"
class="com.howtodoinjava.BeanClass"
scope="thread" />
```

Spring Bean Java Config

Spring Bean Java Config

1. Create bean with component scanning enabled
2. Create bean using @Bean annotation

Spring Bean Java Config

```
@Configuration
@ComponentScan(basePackages = “org.cap.service")
public class AppConfig {


}
```

```
@Configuration
public class AppConfig {

    @Bean
    public EmployeeManager employeeManager() {
        return new EmployeeManagerImpl();
    }

}
```

Spring Bean XML Config

Spring Bean XML Config

Same as Spring4 version with import resources.

```
<beans>


   <import resource="employee.xml"/>
   <import resource="department.xml"/>
</beans>
```

Eager and Lazy Load

Eager and Lazy Load

```
<beans>
 <bean id="employeeManager"
class=“org.cap.service.impl.EmployeeManagerImpl“  lazy-init="true"/>
 <beans>
```

```
<beans default-lazy-init="true">

<bean id="employeeManager"
class=“org.cap.service.impl.EmployeeManagerImpl" />

<beans>
```

Eager and Lazy Load

Annotational:

```
import org.springframework.context.annotation.Lazy;

@Configuration
public class AppConfig {

    @Lazy
    @Bean
    public EmployeeManager employeeManager() {
        return new EmployeeManagerImpl();
    }

}
```

Dispatcher Servlet

DispatcherServlet

BEAN	RESPONSIBILITIES
HandlerMapping	Maps incoming web requests to handlers and pre- and post-processors
HandlerAdapter	Invokes the handler which resolves arguments and dependencies, such as annotated arguments for URL-mapped controller method endpoints
HandlerExceptionResolver	Allows programmatic handling of exceptions and maps exceptions to views
ViewResolver	Resolves logical view names to view instances
LocaleResolver	Resolves the client’s locale in order to enable internationalization
LocaleContextResolver	A richer extension of LocaleResolver, with timezone information
ThemeResolver	Resolves themes configured in your app for enhanced user experience
MultipartResolver	Handles multipart file uploads as part of HTTP requests
FlashMapManager	Manages FlashMap instances that store temporary Flash attributes between requests redirected from one another

Spring CORS

Spring CORS

ATTRIBUTE	DESCRIPTION
origins	List of allowed origins. It’s value is placed in the Access-Control-Allow-Origin header of both the pre-flight response and the actual response. **– * – means that all origins are allowed.** **– If undefined, all origins are allowed.**
allowedHeaders	List of request headers that can be used during the actual request. Value is used in preflight’s response header Access-Control-Allow-Headers. **– * – means that all headers requested by the client are allowed.** **– If undefined, all requested headers are allowed.**
methods	List of supported HTTP request methods. If undefined, methods defined by RequestMapping annotation are used.
exposedHeaders	List of response headers that the browser will allow the client to access. Value is set in actual response header Access-Control-Expose-Headers. – If undefined, an empty exposed header list is used.
allowCredentials	It determine whether browser should include any cookies associated with the request. – false – cookies should not included. – "" (empty string) – means *undefined*. – true – pre-flight response will include the header Access-Control-Allow-Credentials with value set to true. – If undefined, credentials are allowed.
maxAge	maximum age (in seconds) of the cache duration for pre-flight responses. Value is set in header Access-Control-Max-Age. – If undefined, max age is set to 1800 seconds (30 minutes).

Spring CORS

```
@CrossOrigin(origins = "*", allowedHeaders = "*")
@Controller
public class HomeController
{
    @GetMapping(path="/")
    public String homeInit(Model model) {
        return "home";
    }
}
```

```
@Controller
public class HomeController
{
    @CrossOrigin(origins = "*", allowedHeaders = "*")
    @GetMapping(path="/")
    public String homeInit(Model model) {
        return "home";
    }
}
```

@CrossOrgin Overridden at Method Level

```
@Controller
@CrossOrigin(origins = "*", allowedHeaders = "*")
public class HomeController
{
    @CrossOrigin(origins = "http://example.com")
    @GetMapping(path="/")
    public String homeInit(Model model) {
        return "home";
    }
}
```

SpringBoot CORS with WebMvcConfigurer

```
@Configuration
@EnableWebMvc
public class CorsConfiguration extends
WebMvcConfigurerAdapter
{
    @Override
    public void addCorsMappings(CorsRegistry registry) {
        registry.addMapping("/**")
                .allowedMethods("GET", "POST");
    }
}
```

CORS with Spring Security

```
@EnableWebSecurity
public class WebSecurityConfig extends WebSecurityConfigurerAdapter {

    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http.cors().and()
            //other config
    }

    @Bean
    CorsConfigurationSource corsConfigurationSource()
    {
        CorsConfiguration configuration = new CorsConfiguration();
        configuration.setAllowedOrigins(Arrays.asList("https://example.com"));
        configuration.setAllowedMethods(Arrays.asList("GET","POST"));
        UrlBasedCorsConfigurationSource source = new UrlBasedCorsConfigurationSource();
        source.registerCorsConfiguration("/**", configuration);
        return source;
    }
}
```

Summary

Scopes

Spring Bean Java Config

Spring Bean XML Config

Dispatcher Servlet

Spring CORS

THANK YOU

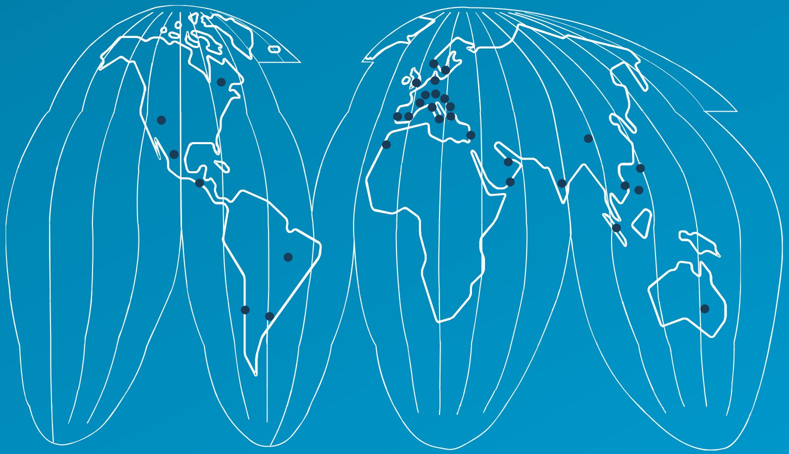
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

People matter, results count.

About Capgemini

With more than 145,000 people in 40 countries, Capgemini is one of the world's foremost providers of consulting, technology and outsourcing services. The Group reported 2014 global revenues of EUR 10.5 billion.
Together with its clients, Capgemini creates and delivers business and technology solutions that fit their needs and drive the results they want. A deeply multicultural organization, Capgemini has developed its own way of working, the Collaborative Business Experience™, and draws on Rightshore®, its worldwide delivery model.

Rightshore® is a trademark belonging to Capgemini



www.capgemini.com

The information contained in this presentation is proprietary.