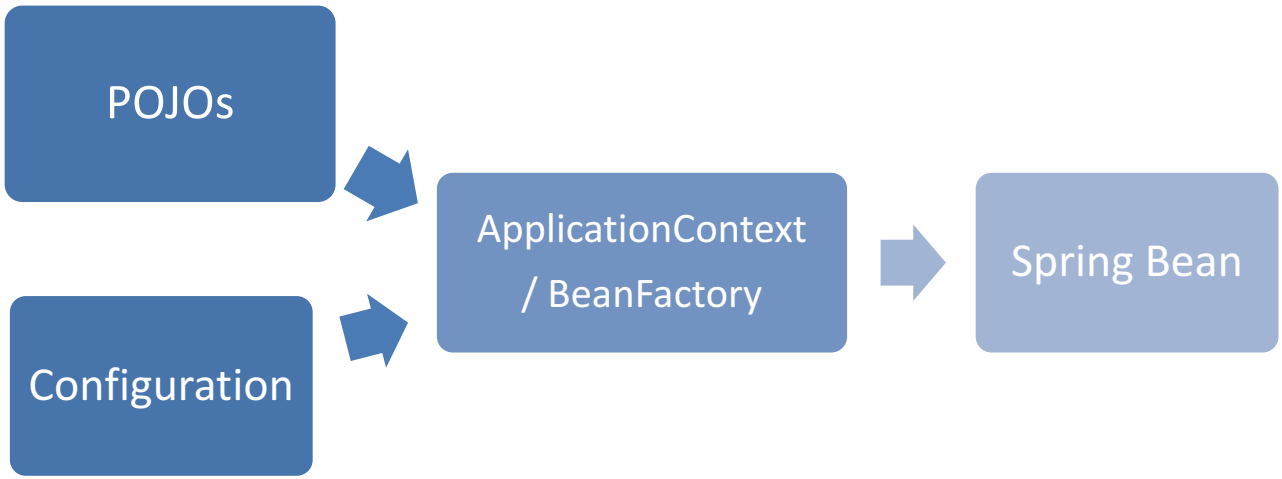
Defining Spring Beans

ApplicationContext

Application Context



Creating an ApplicationContext

Simple

```
new AnnotationConfigApplicationContext(AppConfig.class);
```

Programmaticaly

```
context = new AnnotationConfigApplicationContext();
context.register(AppConfig.class);
…
context.refresh();
```

Scanning

```
context = new AnnotationConfigApplicationContext();
context.scan("com.patbaumgartner.app");
context.refresh();
```

Java Config

Instantiating Bean

```
@Configuration
public class AppRepositoryConfig {

  @Bean(name = "accountRepository", initMethod = "init")
  public AccountRepository accountRepository() {
    return new AccountMapRepository();
  }

  @Bean
  public AuditRepository auditRepository() {
    return new AuditMapRepository();
  }
}
```

@PropertySource

```
@Configuration
@PropertySource("classpath:jdbc.properties")
public class AppConfig {

    @Value("${jdbc.url}")
    private String jdbcUrl;

    @Bean
    public DataSourse dataSource() {
        return new SimpleDataSource(jdbcUrl);
    }
}
```

Dependency Injection

```
@Configuration
public class ExchangerServiceConfig {

  @Autowired
  private CurrencyRepository currencyRepository;

  @Bean
  public CurrencyService currencyService() {
    return new CurrencyServiceImpl(currencyRepository);
  }

  @Bean(name = {"orderBuilder", "builder"})
  public OrderBuilder orderBuilder() {
    return new OrderBuilder(currencyService(), accountService());
  }
}
```

@Import

```
@Configuration
@Import({RepositoryConfig.class, ServiceConfig.class})
public class AppConfig {
}

@Configuration
public class RepositoryConfig { … }

@Configuration
public class ServiceConfig { … }
```

Annotation based Configuration

Annotation based Configuration

- An alternative to XML and Java Config setups
- Rapid prototyping

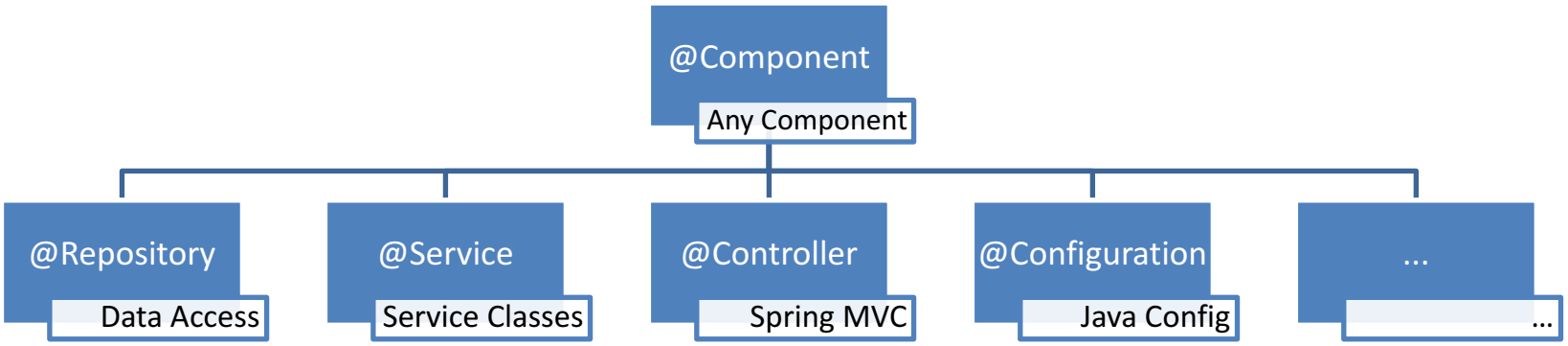
Basic annotations (1)

- Spring Annotations
  - @Autowired
  - @Qualifier
  - @Required
  - @Value
- JSR 250
  - @Resource
  - @PostConstruct
  - @PreDestroy

Basic annotations (2)

- Context
  - @Scope
  - @Bean
  - @DependsOn
  - @Lazy
- Transactional
  - @Transactional
- …

Stereotype Annotations



Basic configuration

```
@Component
public class ApiAuthenticationProvider implements
    AuthenticationProvider {
    …
}
```

```
@ComponentScan(basePackages="com.patbaumgartner.app")
public class ApplicationConfiguration {
    …
}
```

@Value

```
@Component
public class ApiAuthenticationProvider implements
    AuthenticationProvider {

    @Value("${provider.api.url}")
    private String apiUrl;

    @Value("${provider.api.username}")
    private String apiUsername;

    @Value("${provider.api.password}")
    private String apiPassword;

    …
}
```

@Autowired

```
@Service("accountService")
public class AccountServiceImpl implements AuthenticationProvider {

   @Autowired
   private AccountRepository repository;

   @Autowired
   public AccountServiceImpl(AccountRepository repository) { … }

   @Autowired(required = false)
   public void setRepository(AccountRepository repository) { … }

   @Autowired
   public void populate(Repository r, OrderService s) { … }
 }
```

@Qualifier

```
@Service
public class AccountReportService implements ReportService {
    @Autowired
    @Qualifier("mainDataSource")
    private DataSource mainDS;

    @Autowired
    @Qualifier("accountDataSource")
    private DataSource accountDS;

    …
}
```

No @Qualifier needed - Fallback

```
@Service
public class AccountReportService implements ReportService {
    @Autowired
    private DataSource mainDataSource;

    @Autowired
    private DataSource accountDataSource;

    …
}
```