Quarkus Cheat Sheet

Native · CDI · Panache · Reactive · Dev Mode

worksheetdownload.com

{ }

SETUP & CLI

```
quarkus create app com.example:my-app
--extension=resteasy-reactive,hibernate-orm-panache
mvn quarkus:dev  (live reload)
mvn package -Dnative  (GraalVM native
image)
quarkus extension add smallrye-health
```

REST ENDPOINTS

```
@Path("/items") @ApplicationScoped
public class ItemResource {
@GET @Produces(APPLICATION_JSON)
public List<Item> list(){ return
Item.listAll(); }
@POST
@Transactional
public void create(Item i){ i.persist();
}
}
```

CDI BEANS

```
@ApplicationScoped / @RequestScoped /
@Singleton
@Inject MyService svc;
@ConfigProperty(name="greeting") String
greeting;
@Produces  for custom beans
```

CDI context is compile-time no
runtime scanning

```
application.properties for all config
```

PANACHE ENTITIES

```
@Entity public class Item extends
PanacheEntity{
public String name;
public static List<Item>
findByName(String n){
return list("name",n);
}
}
Item.findAll() / Item.count() /
Item.listAll()
```

REACTIVE PROGRAMMING

```
@GET @Path("/{id}")
public Uni<Item> getItem(@PathParam Long
id){
return Item.findById(id);
}
Multi<Item> stream = Item.streamAll();
<dep>quarkus-resteasy-reactive +
hibernate-reactive-panache</dep>
```

NATIVE IMAGE BUILD

```
mvn package -Pnative  (requires GraalVM)
mvn package -Pnative
-Dquarkus.native.container-build=true
```

Results in ./target/my-app-runner
binary
Startup <50ms; reduced memory
footprint
Limitations: reflection, dynamic
proxies need registration

```
@RegisterForReflection on needed classes
```

CONFIGURATION

```
# application.properties
greeting=Hello
%prod.greeting=Hi there
@ConfigProperty(name="greeting",defaultValue="Hey")
String greeting;
```

Profiles: dev / test / prod
auto-applied

TESTING

```
@QuarkusTest
class ItemResourceTest {
@Test void testList(){
given().when().get("/items")
.then().statusCode(200);
}
}
@QuarkusIntegrationTest for native
testing
```

HEALTH CHECKS

```
<dep>quarkus-smallrye-health</dep>
@Liveness / @Readiness
public class MyCheck implements
HealthCheck{
public HealthCheckResponse call(){
return HealthCheckResponse.up("db");
}
}
GET /q/health/live  /q/health/ready
```

MESSAGING (KAFKA)

```
@Incoming("prices") void process(double
price){}
@Outgoing("prices-out") Multi<Double>
generate(){}
<dep>quarkus-smallrye-reactive-messaging-kafka</dep>
mp.messaging.incoming.prices.topic=prices-topic
mp.messaging.incoming.prices.bootstrap.servers=...
```

COMMON GOTCHAS

CDI beans are resolved at build time
runtime classloading fails
Native: reflection-heavy libs need
@RegisterForReflection
Hibernate Reactive: can't mix
blocking and reactive sessions
Dev mode hot reload works for config
and code, not schema
Extension versions must match
Quarkus BOM version