



BETTER PROJECTS
FASTER

BPF.LI/S

KARSTEN SILZ

9. NOVEMBER 2021

TECHNOLOGIEN UND TOOLS MIT JHIPSTER SCHNELLER AUSWÄHLEN



FOLIEN, TUTORIAL & MEHR

BPF.LI/PIE

TECHNOLOGIEN & TOOLS MIT JHIPSTER SCHNELLER AUSWÄHLEN

JAVA: MEHR TECHNOLOGIEN ÖFTER WÄHLEN

WIE WÄHLEN WIR TECHNOLOGIEN AUS?

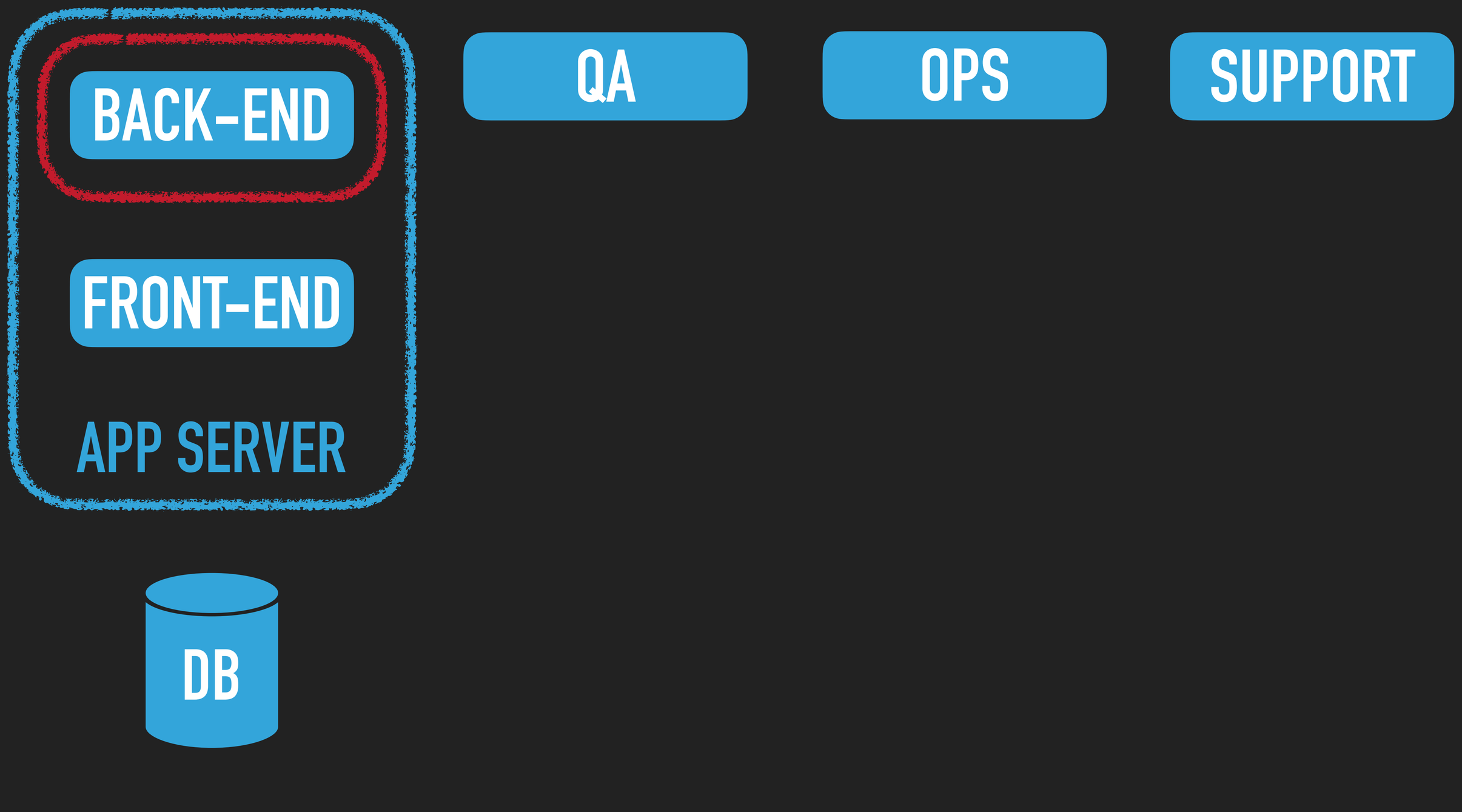
WAS IST JHIPSTER?

WIE HILFT JHIPSTER?

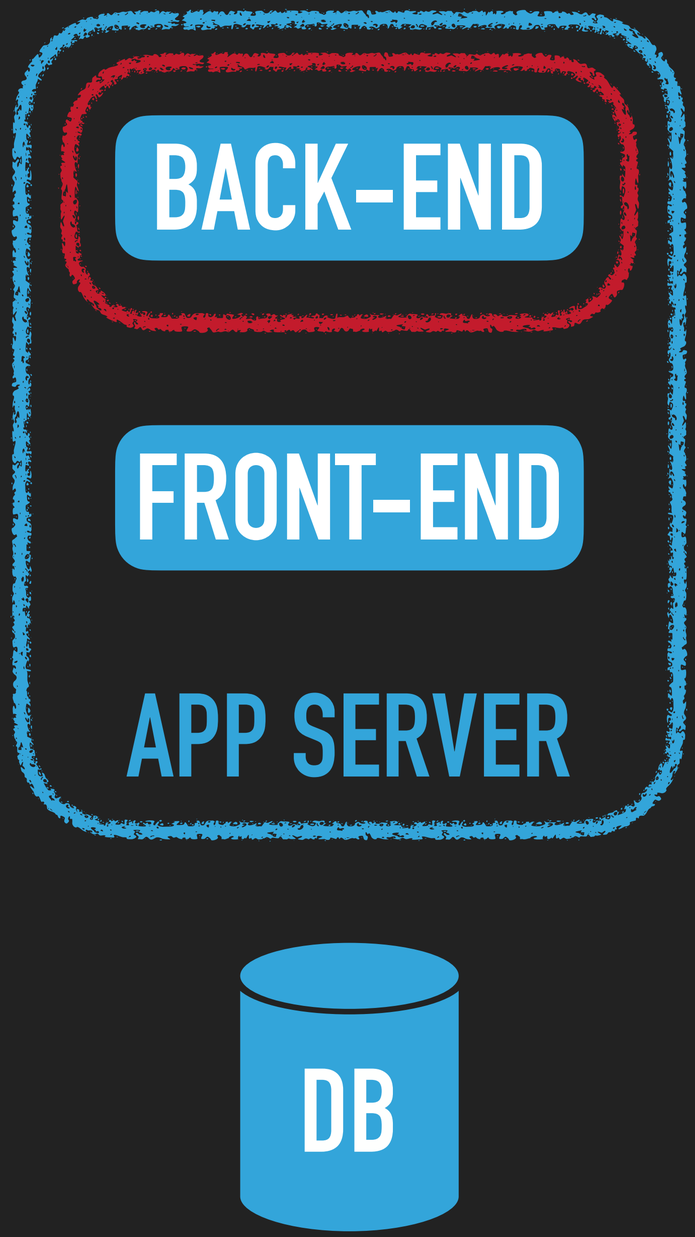
MEHR TECHNOLOGIEN
ÖFTER WÄHLEN

MEHR TECHNOLOGIEN WÄHLEN

FRÜHER


BACK-END
FRONT-END
APP SERVER
DB
QA
OPS
SUPPORT

ÄNDERUNGEN



QA

OPS

SUPPORT

MICROSERVICES & CONTAINER

BACK-END

FRONT-END

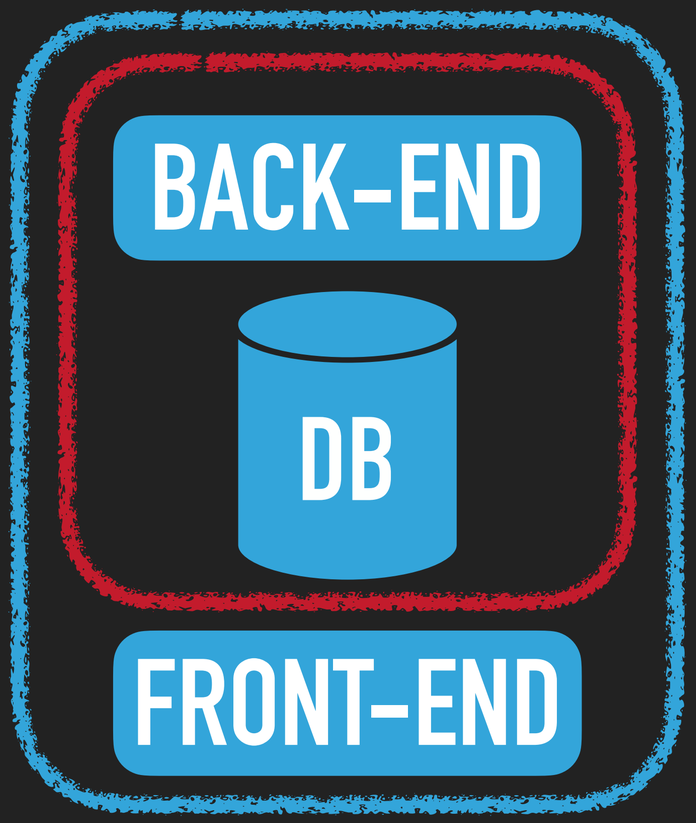
QA

OPS

SUPPORT



MICROSERVICES & CONTAINER



QA

OPS

SUPPORT

MICROSERVICES & CONTAINER

BACK-END

QA

OPS

SUPPORT

FRONT-END

BACK-END

QA

OPS

SUPPORT

FRONT-END

FULL-STACK DEVELOPER

BACK-END

FRONT-END

QA

OPS

SUPPORT

FULL-STACK DEVELOPER

BACK-END

FRONT-END

QA

OPS

SUPPORT

SHIFT LEFT

BACK-END

FRONT-END

QA

OPS

SUPPORT

SHIFT LEFT

BACK-END

FRONT-END

QA

OPS

SUPPORT

DEVOPS

BACK-END

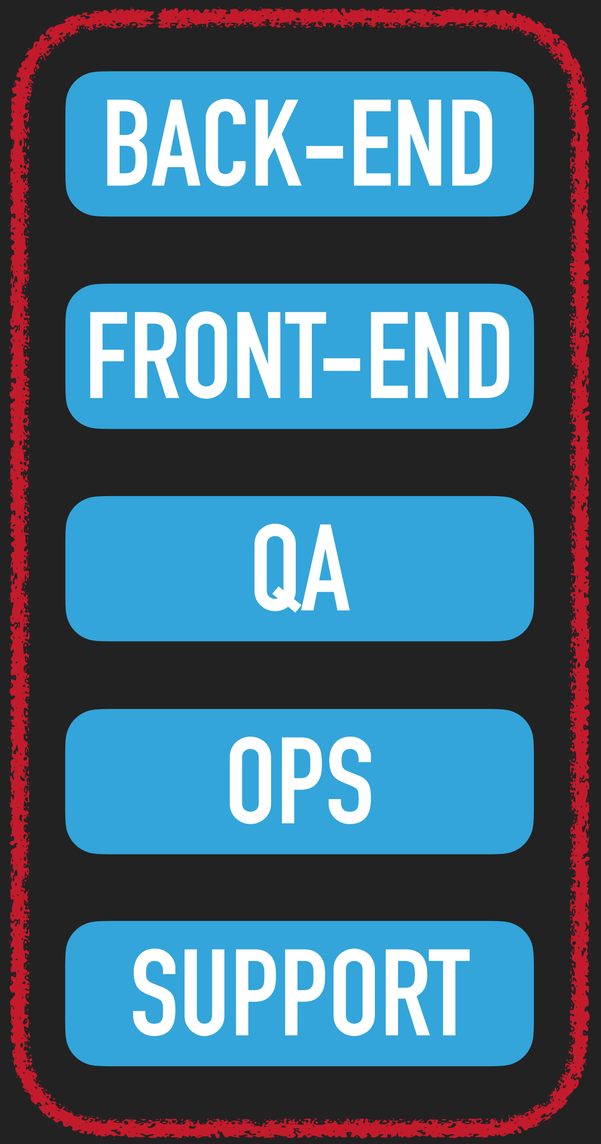
FRONT-END

QA

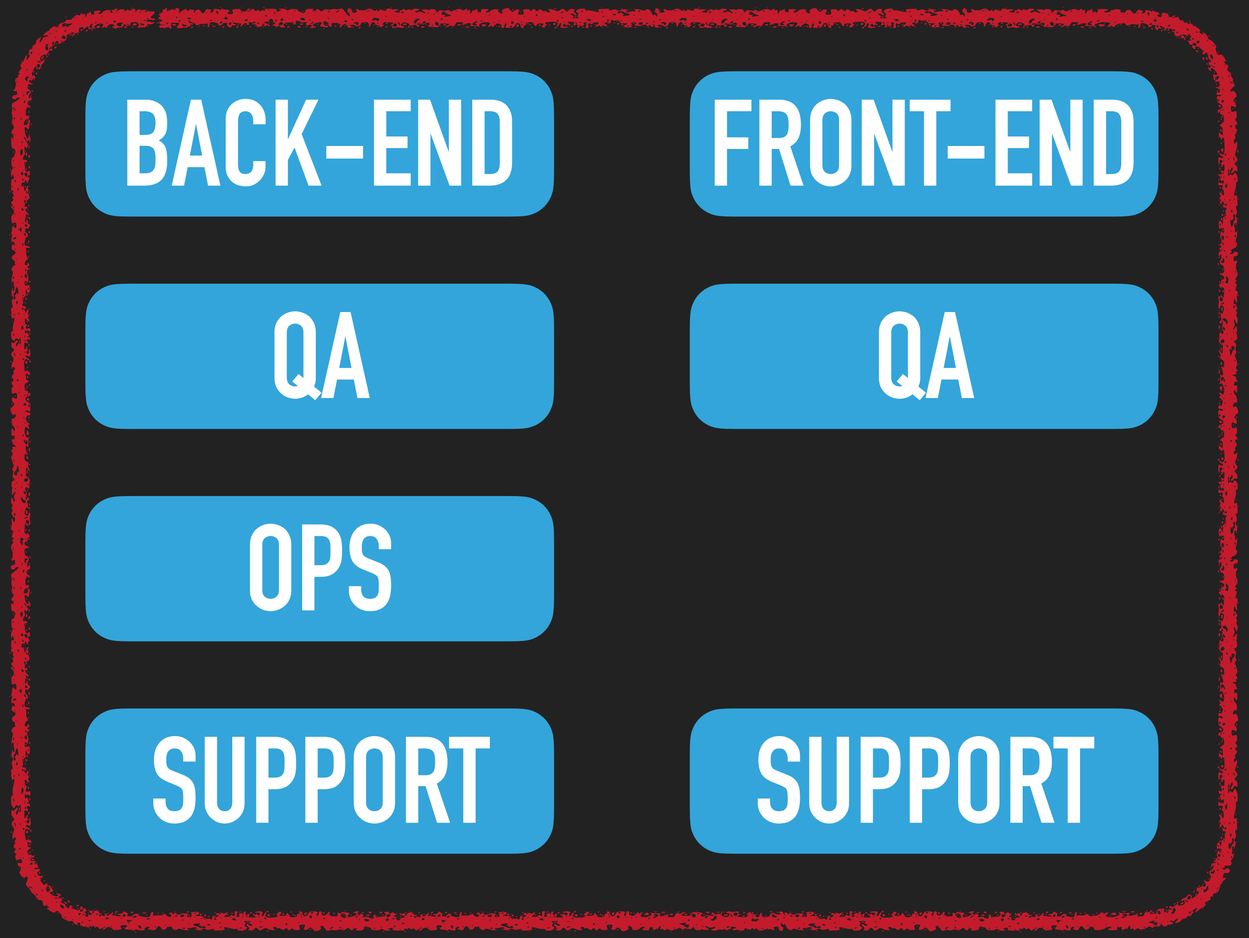
OPS

SUPPORT

DEVOPS



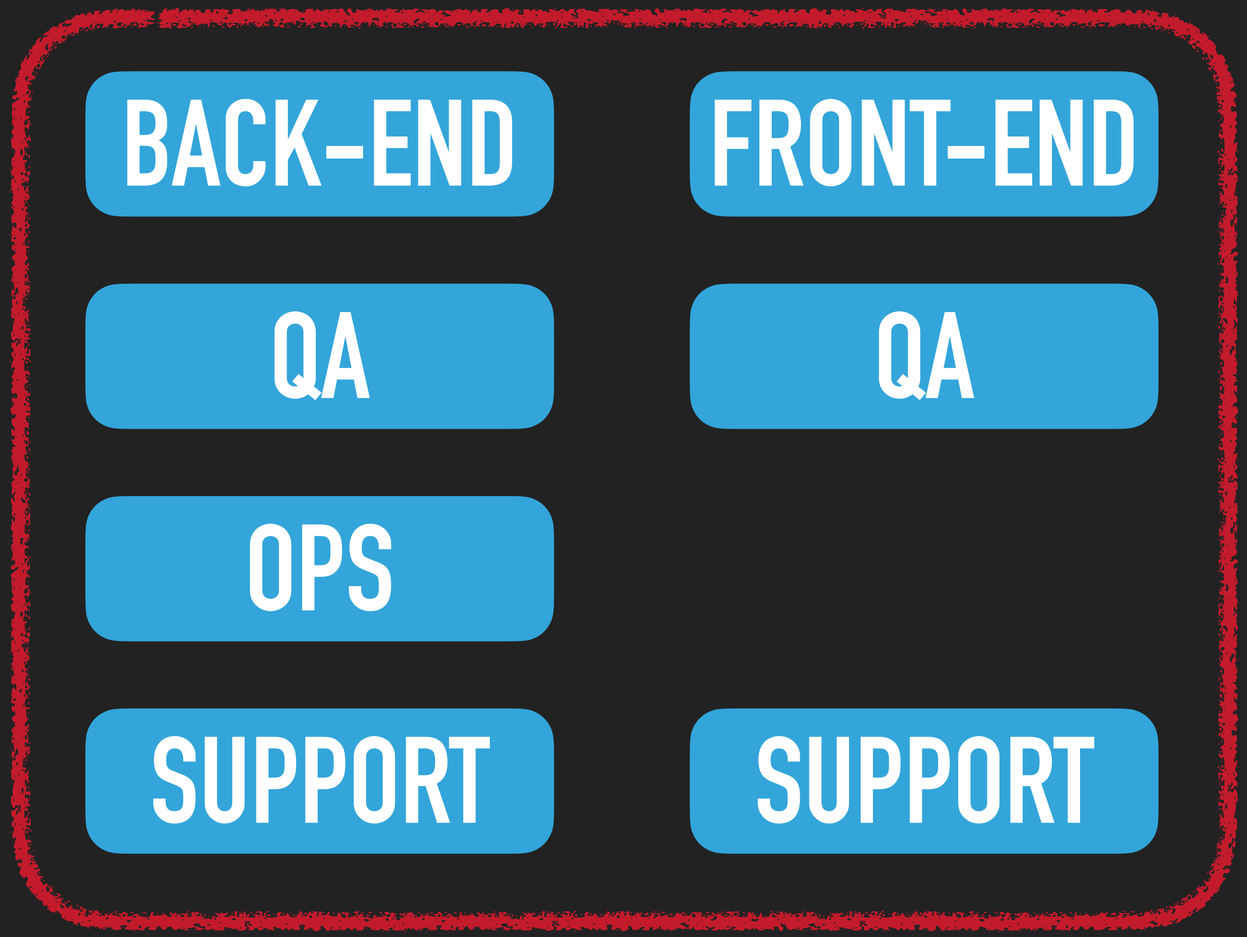
SINGLE PAGE APPLICATIONS



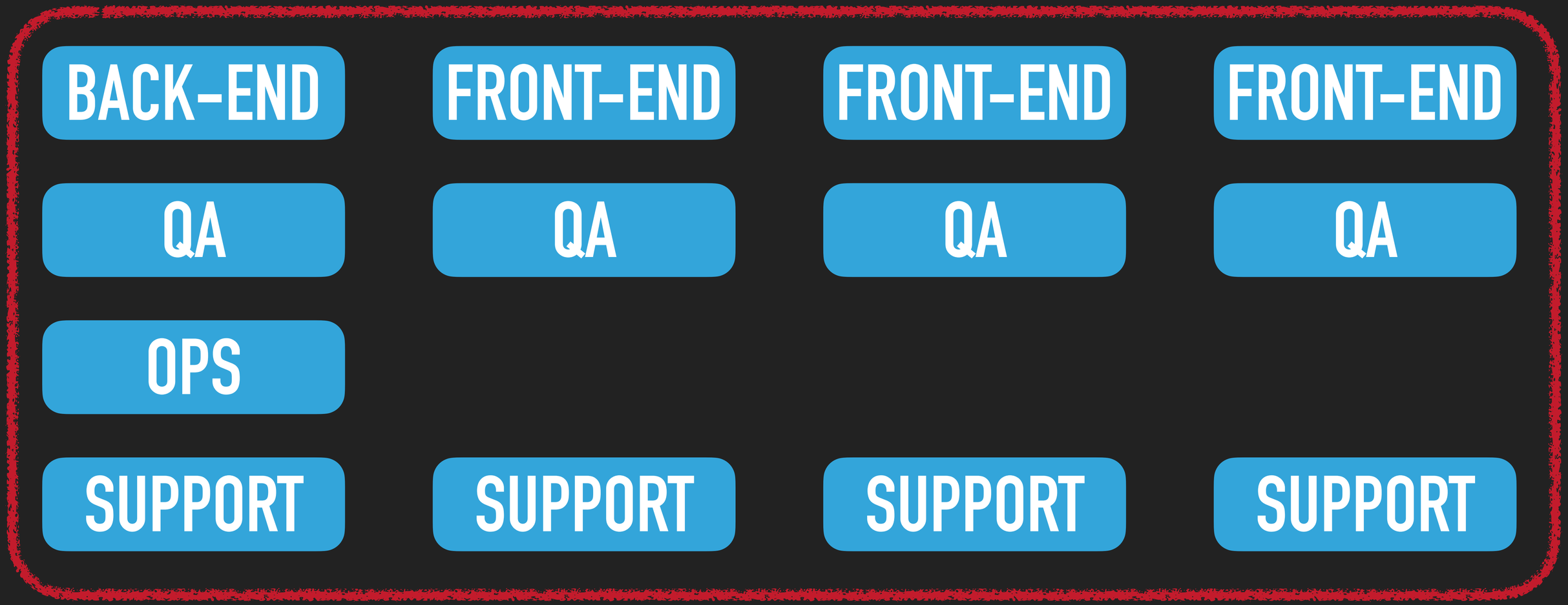
JAVA

JS

SINGLE PAGE APPLICATIONS



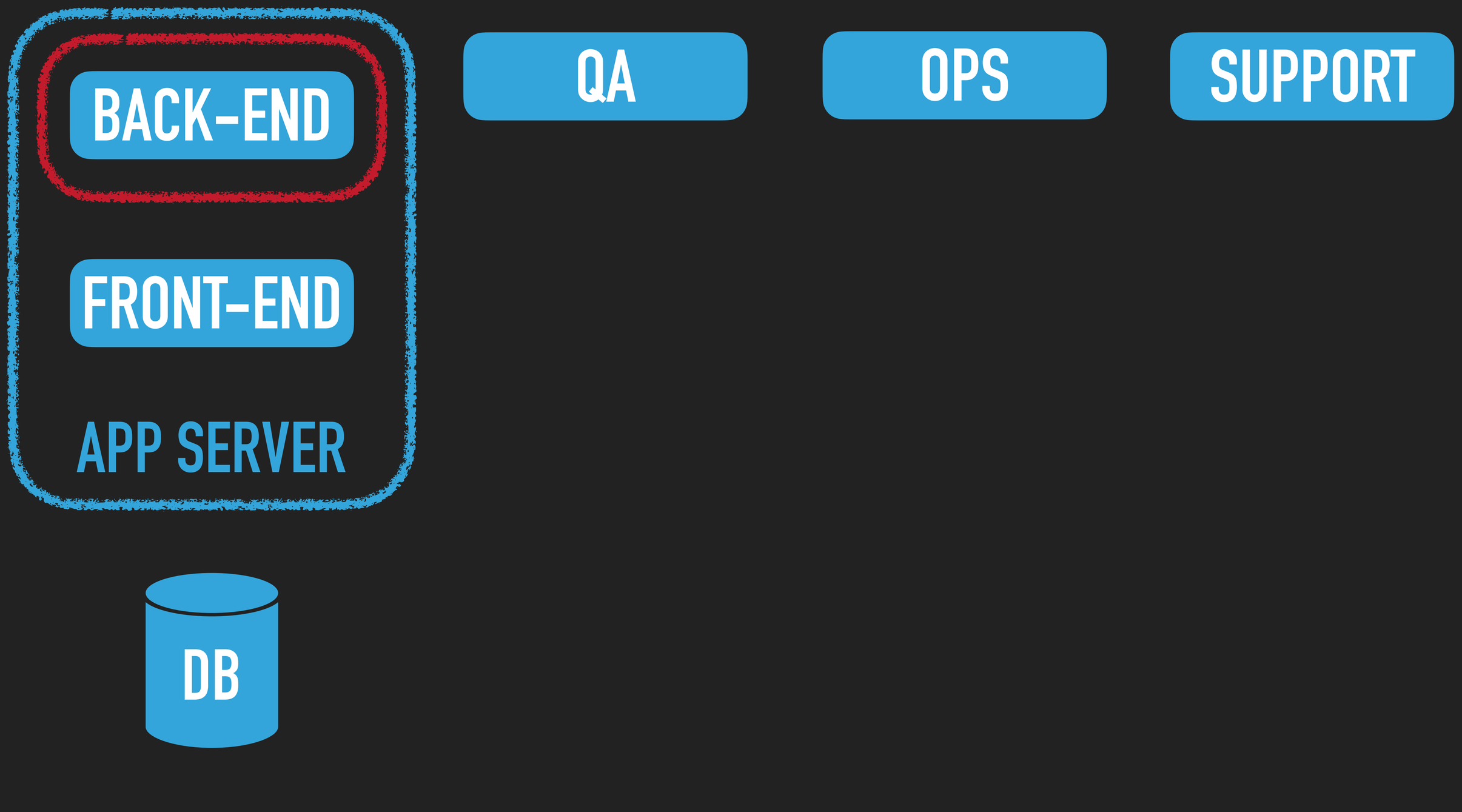
JAVA JS MOBILE APPS



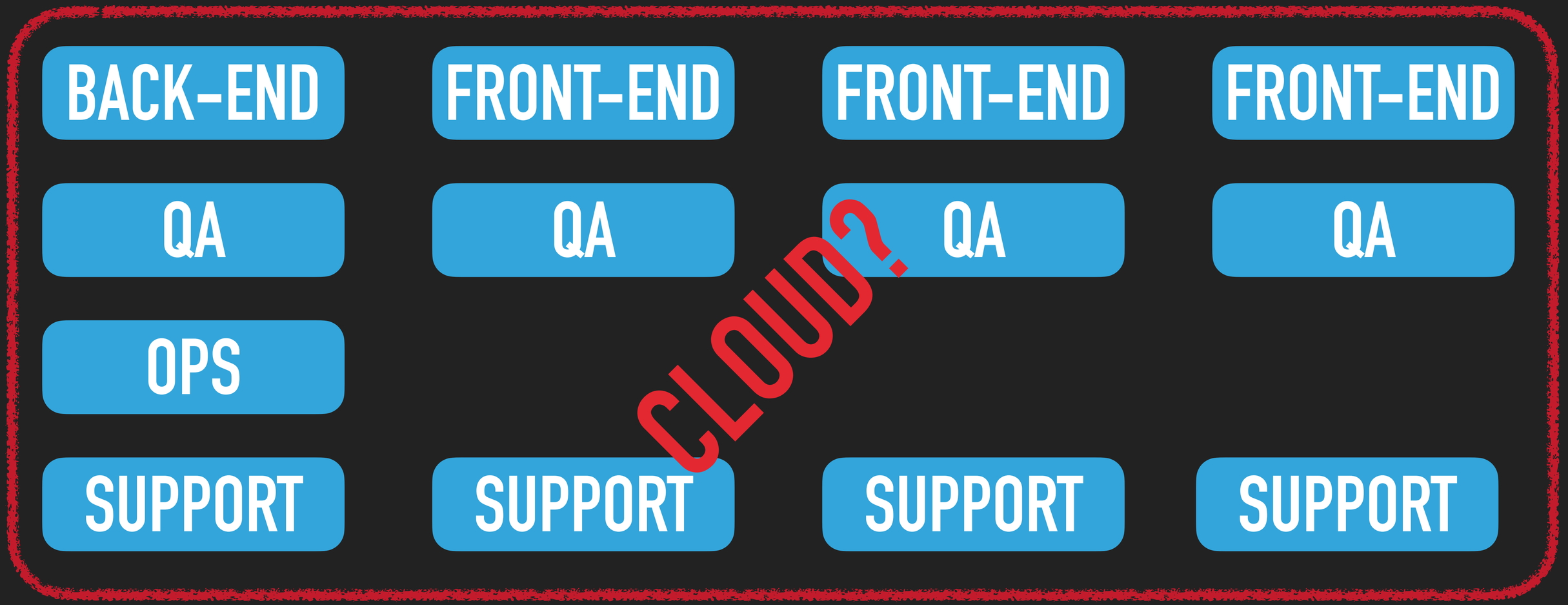
JAVA JS IOS ANDR.

AUßERDEM:
WOFÜR NEHMEN WIR
CLOUD-KOMPONENTEN?

FRÜHER


BACK-END
FRONT-END
APP SERVER
DB
QA
OPS
SUPPORT

HEUTE



JAVA JS IOS ANDR.

UNSERE PROJEKTE: IRGENDWO ZWISCHEN
"FRÜHER" UND "HEUTE"

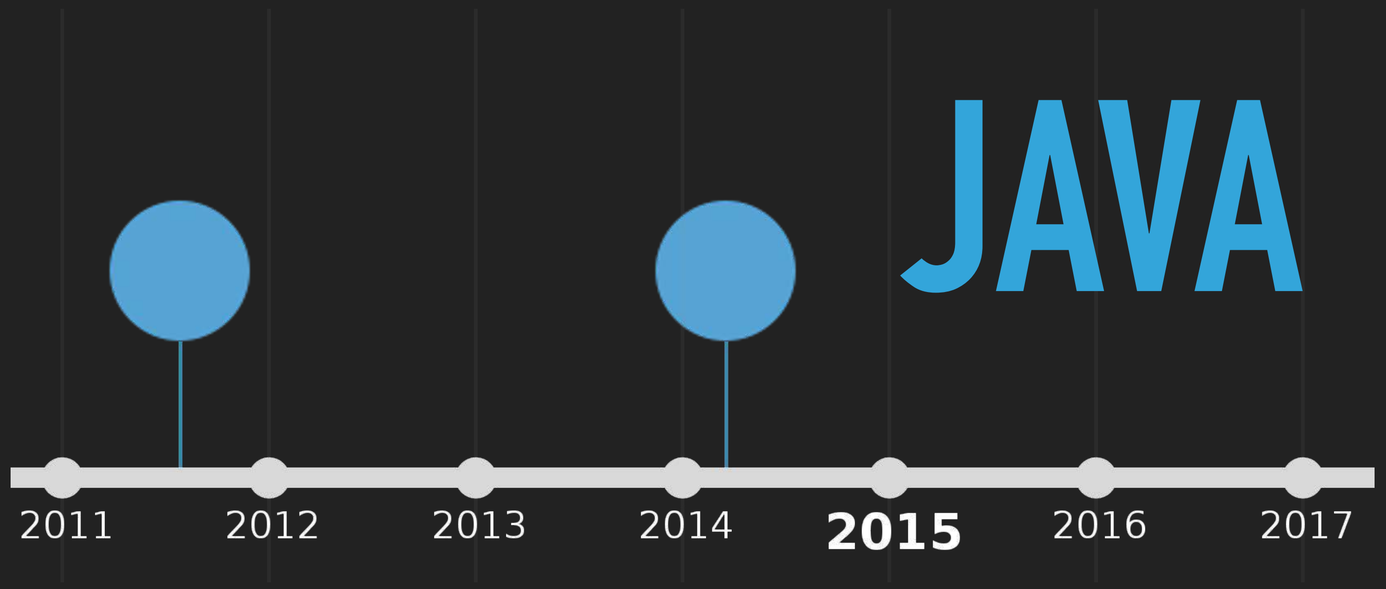
MEIN PROJEKT: "HEUTE"

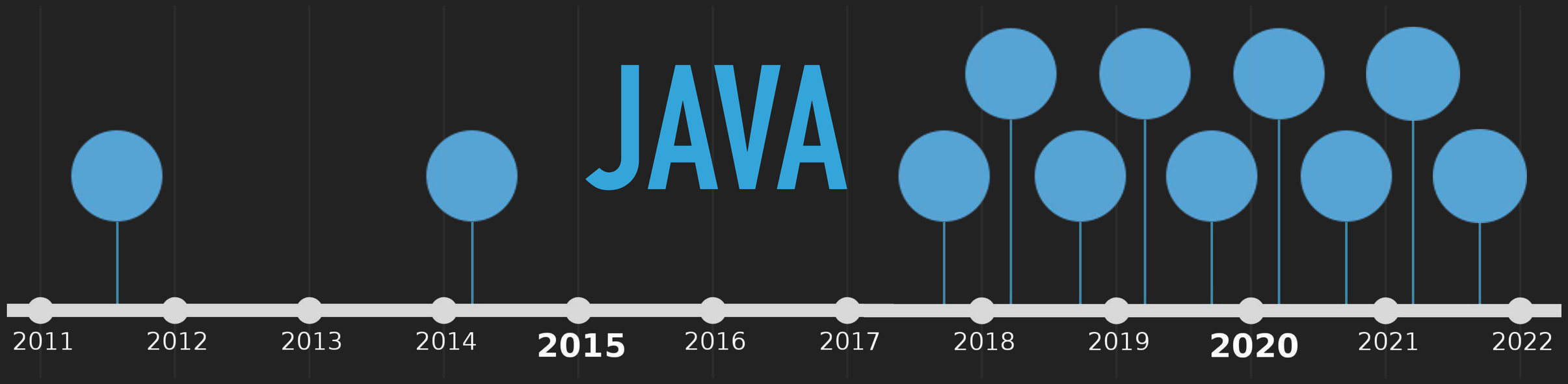
MEHR TECHNOLOGIEN

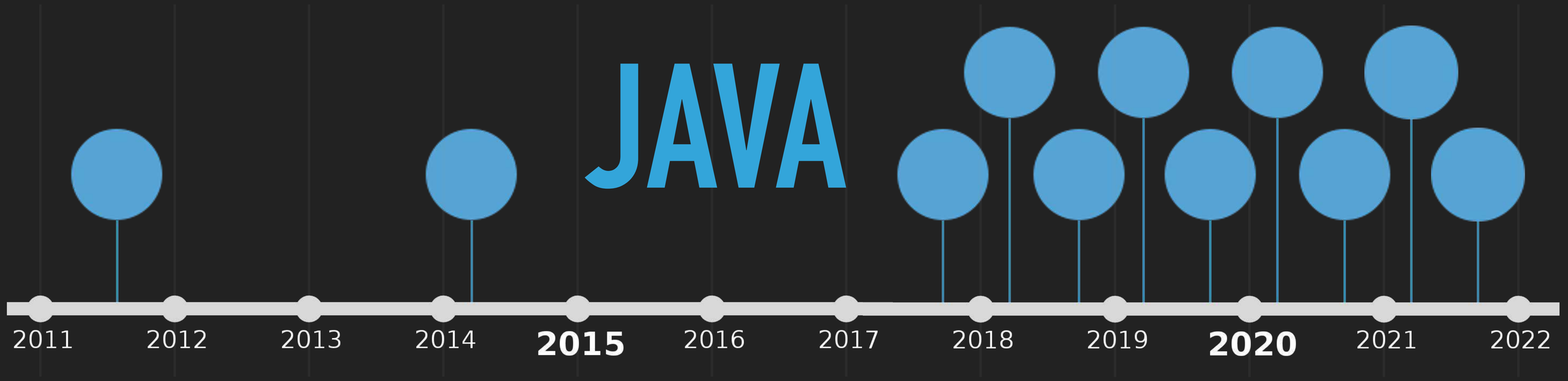
MEHR TECHNOLOGIEN
ÖFTER WÄHLEN

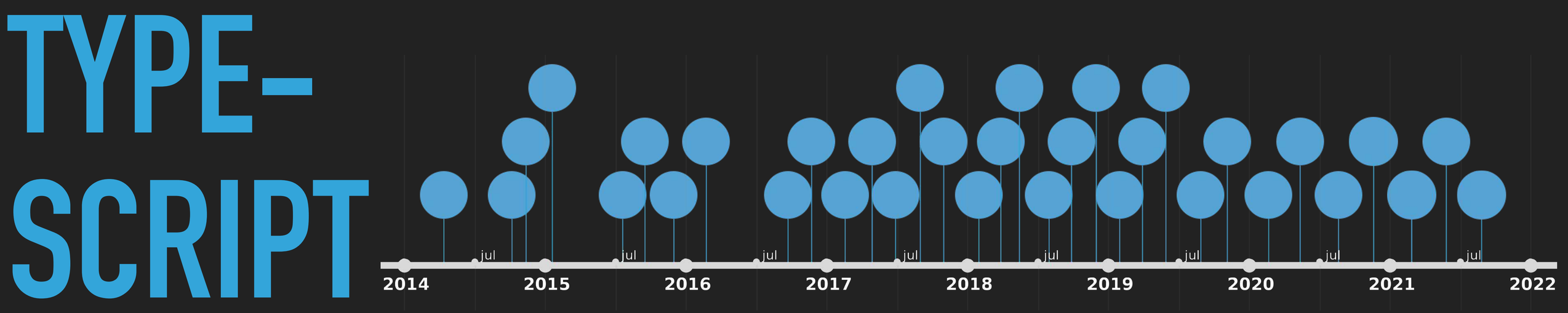
ÖFTER TECHNOLOGIEN WÄHLEN

MEHR RELEASES


JAVA
2011
2012
2013
2014
2015
2016
2017


JAVA
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022


JAVA
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022


TYPE-
SCRIPT
jul
jul
jul
jul
jul
jul
jul
jul
2014
2015
2016
2017
2018
2019
2020
2021
2022

MEHR ALTERNATIVEN IN JAVA

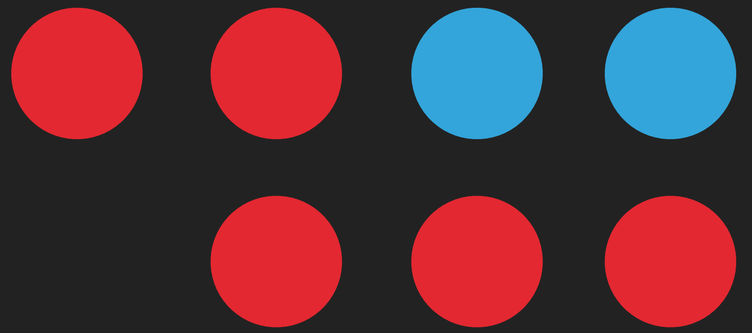
FRAMEWORKS



FRÜHER

FRAMEWORKS





FRÜHER

HEUTE

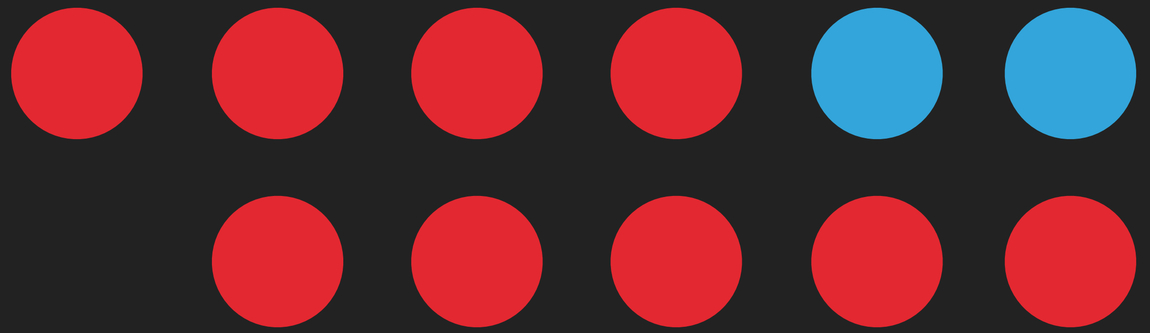
JAVA-DISTRIBUTIONEN



FRÜHER

JAVA-DISTRIBUTIONEN





FRÜHER

HEUTE

MEHR TECHNOLOGIEN
ÖFTER WÄHLEN

~~JAVA: MEHR TECHNOLOGIEN ÖFTER WÄHLEN~~

WIE WÄHLEN WIR TECHNOLOGIEN AUS?

WAS IST JHIPSTER?

WIE HILFT JHIPSTER?

WARUM MIR ZUHÖREN – UND GLAUBEN?



22 JAHRE JAVA

JAVA-REPORTER @ INFOQ

2 JHIPSTER-PROJEKTE

VERKAUFE NIX

NUTZE JHIPSTER, SPENDE $4/MONAT

SIE ENTSCHEIDEN!

~~JAVA: MEHR TECHNOLOGIEN ÖFTER WÄHLEN~~

WIE WÄHLEN WIR TECHNOLOGIEN AUS?

WAS IST JHIPSTER?

WIE HILFT JHIPSTER?

VORAUSWAHL

GENAUE PRÜFUNG

WIE WÄHLEN WIR TECHNOLOGIEN AUS?

KANDIDATEN SIEBEN

FUNKTIONIERT?

SKALIERT?

POPULÄR?

ERWEITERBAR?

SICHER?

RISIKO?

COMPLIANCE?

VORAUSWAHL

UNBEKANNTE TECHNOLOGIEN!

WELCHE KANDIDATEN?

WIE BEWERTEN?

WIE AUSSIEBEN?

VORAUSWAHL: PROBLEME

VORAUSWAHL

GENAUE PRÜFUNG

WIE WÄHLEN WIR TECHNOLOGIEN AUS?

UNSERE UMGEBUNG

UNSERE DATEN

UNSERE IDE

BUILD & DEPLOY

DEBUG

TEST

CI/CD

MONITORING

GENAUE PRÜFUNG

UNBEKANNTE TECHNOLOGIEN!

WIE PROGRAMM ERSTELLEN?

UNSERE DATEN?

GENAUE PRÜFUNG: PROBLEME

VORAUSWAHL

GENAUE PRÜFUNG

SO WÄHLEN WIR TECHNOLOGIEN AUS!

~~JAVA: MEHR TECHNOLOGIEN ÖFTER WÄHLEN~~

~~WIE WÄHLEN WIR TECHNOLOGIEN AUS?~~

WAS IST JHIPSTER?

WIE HILFT JHIPSTER?

WAS IST JHIPSTER?



SUPERSTAR-PROGRAMMIERER

KENNT ALLES

KONFIGURIERT ALLES

SCHREIBT VIEL CODE

JHIPSTER IST..



OPEN-SOURCE CODE-GENERATOR

JAVA, NODE.JS, GIT

KEINE PROPRIETÄRE SCHICHT

PRODUKTIONS-REIFER CODE

JHIPSTER

LIVE DEMO

APPLIKATION
CRUD-CODE

JHIPSTER-WIZARD



APPLIKATION & CRUD-CODE GENERIERT

JHIPSTER: GLEICHE APPLIKATION MIT UNTERSCHIEDLICHEN TECHNOLOGIEN!

SEIT 8 JAHREN

REGELMÄßIGE RELEASES (AUßER 2020)

120.000 DOWNLOADS PRO MONAT

GRÜNDER "JAVA ADVOCACY MANAGER" BEI MICROSOFT

SICHER?

TEXT-DATEI ZUR KONFIGURATION
VON APPLIKATION & ENTITIES

JHIPSTER DOMAIN LANUGAGE (JDL)

```
entity Product {
    name String required minlength(4) maxlength(100)
    price BigDecimal required min(0)
    size ProductSize required
    image ImageBlob
    specifications Blob
    description TextBlob
}
```

ENTITY

```
entity Product {
    name String required minlength(4) maxlength(100)
    price BigDecimal required min(0)
    size ProductSize required
    image ImageBlob
    specifications Blob
    description TextBlob
}
```

DATEN-TYPEN

ENTITY

```
entity Product {
    name String required minlength(4) maxlength(100)
    price BigDecimal required min(0)
    size ProductSize required
    image ImageBlob
    specifications Blob
    description TextBlob
}
```

CONSTRAINTS

ENTITY

```
entity Product {
    name String required minlength(4) maxlength(100)
    price BigDecimal required min(0)
    size ProductSize required
    image ImageBlob
    specifications Blob
    description TextBlob
}
```

ENTITY

```
relationship OneToMany {
   ProductCategory{product} to Product{productCategory(name)}
}
```

RELATIONSHIPS

```
entity Product {
    name String required minlength(4) maxlength(100)
    price BigDecimal required min(0)
    size ProductSize required
    image ImageBlob
    specifications Blob
    description TextBlob
}
```

ENTITY

```
relationship OneToMany {
   ProductCategory{product} to Product{productCategory(name)}
}
```

RELATIONSHIPS

```
entity Product {
    name String required minlength(4) maxlength(100)
    price BigDecimal required min(0)
    size ProductSize required
    image ImageBlob
    specifications Blob
    description TextBlob
}
```

ENTITY

```
relationship OneToMany {
   ProductCategory{product} to Product{productCategory(name)}
}
```

RELATIONSHIPS

```
service * with serviceImpl
dto * with mapstruct
paginate * with pagination
```

KONFIGURATION

WIE SUPERSTAR-PROGRAMMIERER

KEINE PROPRIETÄRE SCHICHT

PRODUKTIONS-REIFER CODE

GLEICHE APPLIKATION MIT UNTERSCHIEDLICHEN TECHNOLOGIEN

DAS IST JHIPSTER

~~JAVA: MEHR TECHNOLOGIEN ÖFTER WÄHLEN~~

~~WIE WÄHLEN WIR TECHNOLOGIEN AUS?~~

~~WAS IST JHIPSTER?~~

WIE HILFT JHIPSTER?

WIE HILFT JHIPSTER?

UNBEKANNTE TECHNOLOGIEN!

WELCHE KANDIDATEN?

WIE BEWERTEN?

WIE AUSSIEBEN?

VORAUSWAHL: PROBLEME

UNBEKANNTE TECHNOLOGIEN!

WIE PROGRAMM ERSTELLEN?

UNSERE DATEN?

GENAUE PRÜFUNG: PROBLEME

JHIPSTER: GLEICHE APPLIKATION MIT UNTERSCHIEDLICHEN TECHNOLOGIEN!

VORAUSWAHL: DURCH JHIPSTER

GENAUE PRÜFUNG: JHIPSTER GENERIERT KOMPLETTE APPLIKATION, VON UNBEKANNTEM CODE LERNEN!

WIE HILFT JHIPSTER?

DATENBANKEN: ALLE RELATIONALEN AUSSER DB2;
NOSQL: MONGODB, CASSANDRA, COUCHBASE, NEO4J

FRAMEWORK: SPRING BOOT, QUARKUS, MICRONAUT,
NODE.JS, .NET (TEILWEISE BETA)

FRONT-END: REACT, ANGULAR, VUE.JS

VORAUSWAHL: JHIPSTER

JAVA/KOTLIN, MONOLITH/MICROSERVICES,
BUILD TOOL, AUTHENTIFIZIERUNG, CACHING,
TESTS, CONTINUOUS INTEGRATION,
DEPLOYMENT, MONITORING, . . .

VORAUSWAHL: JHIPSTER

APPLIKATION MIT ALLEN TECHNOLOGIEN GENERIEREN

UNSER DATEN-MODELL ALS JDL IMPORTIEREN

APPLIKATIONEN IN UNSERER UMGEBUNG PRÜFEN

GENAUE PRÜFUNG: JHIPSTER

~~UNSERE UMGEBUNG~~

~~UNSERE DATEN~~

~~UNSERE IDE~~

~~BUILD & DEPLOY~~

~~DEBUG~~

~~TEST~~

~~CI/CD~~

~~MONITORING~~

GENAUE PRÜFUNG: JHIPSTER

. . . UND WIR MÜSSEN DEN JHIPSTER-CODE NOCH NICHT MAL VERSTEHEN!

VON UNBEKANNTEM CODE LERNEN?

SQL VS. NOSQL: ENTITIES

ENTITY:

```
public class Product implements Serializable {

  @Id


  private          id;

  @NotNull
  @Size(min = 2, max = 90)

  private String name;
```

ENTITY: NOSQL

```
@Document(collection = "product")
public class Product implements Serializable {

  @Id



  private String id;

  @NotNull
  @Size(min = 2, max = 90)
  @Field("name")
  private String name;
```

ENTITY: SQL

```
@Entity
@Table(name = "product")
@Cache(usage = CacheConcurrencyStrategy.READ_WRITE)
public class Product implements Serializable {

  @Id
  @GeneratedValue(strategy = GenerationType.SEQUENCE,
    generator = "sequenceGenerator")
  @SequenceGenerator(name = "sequenceGenerator")
  private Long   id;

  @NotNull
  @Size(min = 2, max = 90)
  @Column(name = "name", length = 90, nullable = false, unique = true)
  private String name;
```

ENTITY: SQL

```
@Entity
@Table(name = "product")
@Cache(usage = CacheConcurrencyStrategy.READ_WRITE)
public class Product implements Serializable {

  @Id
  @GeneratedValue(strategy = GenerationType.SEQUENCE,
    generator = "sequenceGenerator")
  @SequenceGenerator(name = "sequenceGenerator")
  private Long   id;

  @NotNull
  @Size(min = 2, max = 90)
  @Column(name = "name", length = 90, nullable = false, unique = true)
  private String name;
```

ANDERER TYP

ENTITY: SQL

```
@Entity
@Table(name = "product")
@Cache(usage = CacheConcurrencyStrategy.READ_WRITE)
public class Product implements Serializable {

  @Id
  @GeneratedValue(strategy = GenerationType.SEQUENCE,
    generator = "sequenceGenerator")
  @SequenceGenerator(name = "sequenceGenerator")
  private Long   id;

  @NotNull
  @Size(min = 2, max = 90)
  @Column(name = "name", length = 90, nullable = false, unique = true)
  private String name;
```

ENTITY: NOSQL

```
@Document(collection = "product")
public class Product implements Serializable {

  @Id



  private String id;

  @NotNull
  @Size(min = 2, max = 90)
  @Field("name")
  private String name;
```

NOSQL & SQL ENTITIES
ÄHNLICH!

ANGULAR VS. REACT

CRUD-CODE:

```
<div>
  <h2 id="         -heading">
    <                      ="jhipsterApp.product.home.title">Products
    </           >
    <         id="jh-create-entity"
                  "btn btn-primary float-right jh-create-entity
                      ">
      <                  icon="plus"          >
      <                      ="jhipsterApp.product.home.createLabel">
        Create new Product
      </           >
    </       >
  </h2>
```

CRUD-CODE: ANGULAR - IMPERATIVE

```
<div>
  <h2 id="page   -heading">
    <span jhiTranslate   ="jhipsterApp.product.home.title">Products
    </span     >
    <button id="jh-create-entity" [routerLink]="['/product/new']"
      class=    "btn btn-primary float-right jh-create-entity
      create-product">
      <fa-icon          icon="plus"</fa-icon>
      <span jhiTranslate   ="jhipsterApp.product.home.createLabel">
        Create new Product
      </span     >
    </button>
  </h2>
```

CRUD-CODE: ANGULAR - IMPERATIVE

PRÄSENTATIONS-CODE IN SEPARATEM TEMPLATE

```
<div>
  <h2 id="page    -heading">
    <span jhiTranslate    ="jhipsterApp.product.home.title">Products
    </span      >
    <button id="jh-create-entity" [routerLink]="['/product/new']"
      class=     "btn btn-primary float-right jh-create-entity
      create-product">
      <fa-icon           icon="plus"</fa-icon>
      <span jhiTranslate    ="jhipsterApp.product.home.createLabel">
        Create new Product
      </span      >
    </button>
  </h2>
```

CRUD-CODE: ANGULAR - IMPERATIVE

STANDARD HTML-TAGS, PROPRIETÄRE ATTRIBUTE

```
<div>
  <h2 id="page-heading">
    <span jhiTranslate   ="jhipsterApp.product.home.title">Products
    </span      >
    <button id="jh-create-entity" [routerLink]="['/product/new']"
      class=     "btn btn-primary float-right jh-create-entity
      create-product">
      <fa-icon          icon="plus"</fa-icon>
      <span jhiTranslate   ="jhipsterApp.product.home.createLabel">
        Create new Product
      </span      >
    </button>
  </h2>
```

CRUD-CODE: ANGULAR - IMPERATIVE

```
<div>
  <h2 id="page   -heading">
    <span jhiTranslate   ="jhipsterApp.product.home.title">Products
    </span     >
    <button id="jh-create-entity" [routerLink]="['/product/new']"
      class=    "btn btn-primary float-right jh-create-entity
      create-product">
      <fa-icon          icon="plus"</fa-icon>
      <span jhiTranslate   ="jhipsterApp.product.home.createLabel">
        Create new Product
      </span     >
    </button>
  </h2>
```

CRUD-CODE: REACT - DECLARATIVE

```
const { productList, match, loading, totalItems } = props;
return (
<div>
  <h2 id="product-heading">
    <Translate contentKey="jhipsterApp.product.home.title">Products
    </Translate>
    <Link   id="jh-create-entity" to={`${match.url}/new`
      className="btn btn-primary float-right jh-create-entity
                  ">
      <FontAwesomeIcon icon="plus"         />
      <Translate contentKey="jhipsterApp.product.home.createLabel">
        Create new Product
      </Translate>
    </Link  >
  </h2>
```

CRUD-CODE: REACT - DECLARATIVE

UI AS CODE - ALLES IN JAVASCRIPT-DATEI

```
const { productList, match, loading, totalItems } = props;
return (
  <div>
    <h2 id="product-heading">
      <Translate contentKey="jhipsterApp.product.home.title">Products
      </Translate>
      <Link   id="jh-create-entity" to={`${match.url}/new`
        className="btn btn-primary float-right jh-create-entity
                  ">
        <FontAwesomeIcon icon="plus"         />
        <Translate contentKey="jhipsterApp.product.home.createLabel">
          Create new Product
        </Translate>
      </Link  >
    </h2>
```

CRUD-CODE: REACT - DECLARATIVE

```
const { productList, match, loading, totalItems } = props;
return (
<div>
  <h2 id="product-heading">
    <Translate contentKey="jhipsterApp.product.home.title">Products
    </Translate>
    <Link   id="jh-create-entity" to={`${match.url}/new`
      className="btn btn-primary float-right jh-create-entity
                    ">
      <FontAwesomeIcon icon="plus"         />
      <Translate contentKey="jhipsterApp.product.home.createLabel">
        Create new Product
      </Translate>
    </Link  >
  </h2>
```

KEINE STANDARD HTML-TAGS

CRUD-CODE: REACT - DECLARATIVE

```
const { productList, match, loading, totalItems } = props;
return (
<div>
  <h2 id="product-heading">
    <Translate contentKey="jhipsterApp.product.home.title">Products
    </Translate>
    <Link   id="jh-create-entity" to={`${match.url}/new`
      className="btn btn-primary float-right jh-create-entity
                  ">
      <FontAwesomeIcon icon="plus"         />
      <Translate contentKey="jhipsterApp.product.home.createLabel">
        Create new Product
      </Translate>
    </Link  >
  </h2>
```

CRUD-CODE: ANGULAR - IMPERATIVE

```
<div>
  <h2 id="page   -heading">
    <span jhiTranslate   ="jhipsterApp.product.home.title">Products
    </span     >
    <button id="jh-create-entity" [routerLink]="['/product/new']"
      class=    "btn btn-primary float-right jh-create-entity
      create-product">
      <fa-icon          icon="plus"</fa-icon>
      <span jhiTranslate   ="jhipsterApp.product.home.createLabel">
        Create new Product
      </span     >
    </button>
  </h2>
```

ANGULAR: IMPERATIVE
REACT: DECLARATIVE

WIR KÖNNEN VON UNBEKANNTEM CODE LERNEN!

LIVE DEMO

APPLIKATION BEGUTACHTEN



APPLIKATION BEGUTACHTEN

TECHNOLOGIE NICHT IN JHIPSTER

WIE PRODUKTIV MIT TECHNOLOGIEN?

WO HILFT JHIPSTER NICHT?

VORAUSWAHL: DURCH JHIPSTER

GENAUE PRÜFUNG: JHIPSTER GENERIERT KOMPLETTE APPLIKATION, VON UNBEKANNTEM CODE LERNEN!

WIE HILFT JHIPSTER?

~~JAVA: MEHR TECHNOLOGIEN ÖFTER WÄHLEN~~

~~WIE WÄHLEN WIR TECHNOLOGIEN AUS?~~

~~WAS IST JHIPSTER?~~

~~WIE HILFT JHIPSTER?~~

ZUSAMMENFASSUNG

WIR ENTWICKLER WÄHLEN
MEHR TECHNOLOGIEN
ÖFTER

VORAUSWAHL

GENAUE PRÜFUNG

WIE WÄHLEN WIR TECHNOLOGIEN AUS?

UNBEKANNTE TECHNOLOGIEN!

WELCHE KANDIDATEN?

WIE BEWERTEN?

WIE AUSSIEBEN?

VORAUSWAHL: PROBLEME

UNBEKANNTE TECHNOLOGIEN!

WIE PROGRAMM ERSTELLEN?

UNSERE DATEN?

GENAUE PRÜFUNG: PROBLEME

WIE SUPERSTAR-PROGRAMMIERER

KEINE PROPRIETÄRE SCHICHT

PRODUKTIONS-REIFER CODE

GLEICHE APPLIKATION MIT UNTERSCHIEDLICHEN TECHNOLOGIEN

DAS IST JHIPSTER

VORAUSWAHL: DURCH JHIPSTER

GENAUE PRÜFUNG: JHIPSTER GENERIERT KOMPLETTE APPLIKATION, VON UNBEKANNTEM CODE LERNEN!

WIE HILFT JHIPSTER?

TECHNOLOGIE NICHT IN JHIPSTER

WIE PRODUKTIV MIT TECHNOLOGIEN?

WO HILFT JHIPSTER NICHT?

TECHNOLOGIEN & TOOLS MIT JHIPSTER SCHNELLER AUSWÄHLEN

WERBUNG

MAI 2022

FULL-STACK JAVA: SPRING BOOT, ANGULAR, FLUTTER

FREIBERUFLER/BEFRISTETE ANSTELLUNG

MILTON KEYNES, LONDON, ODER REMOTE

SUCHE PROJEKT!


The End



FOLIEN

ZUSATZ-INFOS

FEEDBACK

JHIPSTER-TUTORIAL

BPF.LI/PIE