



KARSTEN SILZ, OCT 4, 2022

PICK TECHNOLOGIES FASTER BY CODING WITH JHIPSTER!

WE DEVELOPERS PICK
MORE TECHNOLOGIES
MORE OFTEN

JHIPSTER DOES
PRESELECTION AND HELPS
WITH DETAILED EVALUATION

JHIPSTER GENERATES THE SAME APPLICATION WITH DIFFERENT TECHNOLOGIES AND OUR DATA



SLIDES & MORE

BPF.LI/JET

WHO MADE ME THE EXPERT?



TWO JHIPSTER APPS IN PRODUCTION

INFOQ JAVA EDITOR, PC QCON LONDON

JAVA DEVELOPER FOR 23 YEARS

USE JHIPSTER, JAVA, ANGULAR & FLUTTER – NOT ASSOCIATED

JAVA: PICK MORE TECHNOLOGIES MORE OFTEN

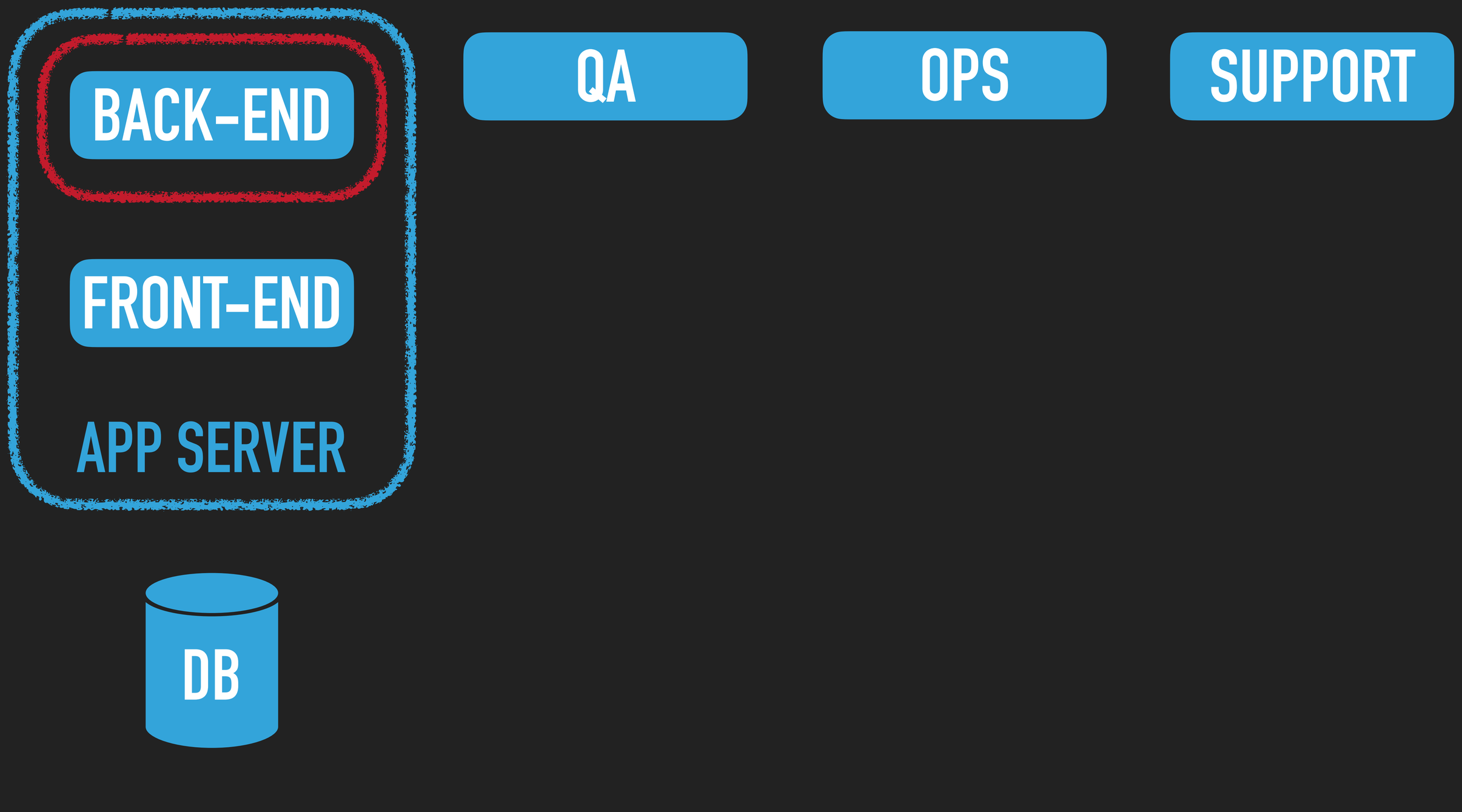
HOW DO WE PICK TECHNOLOGIES?

WHAT IS JHIPSTER?

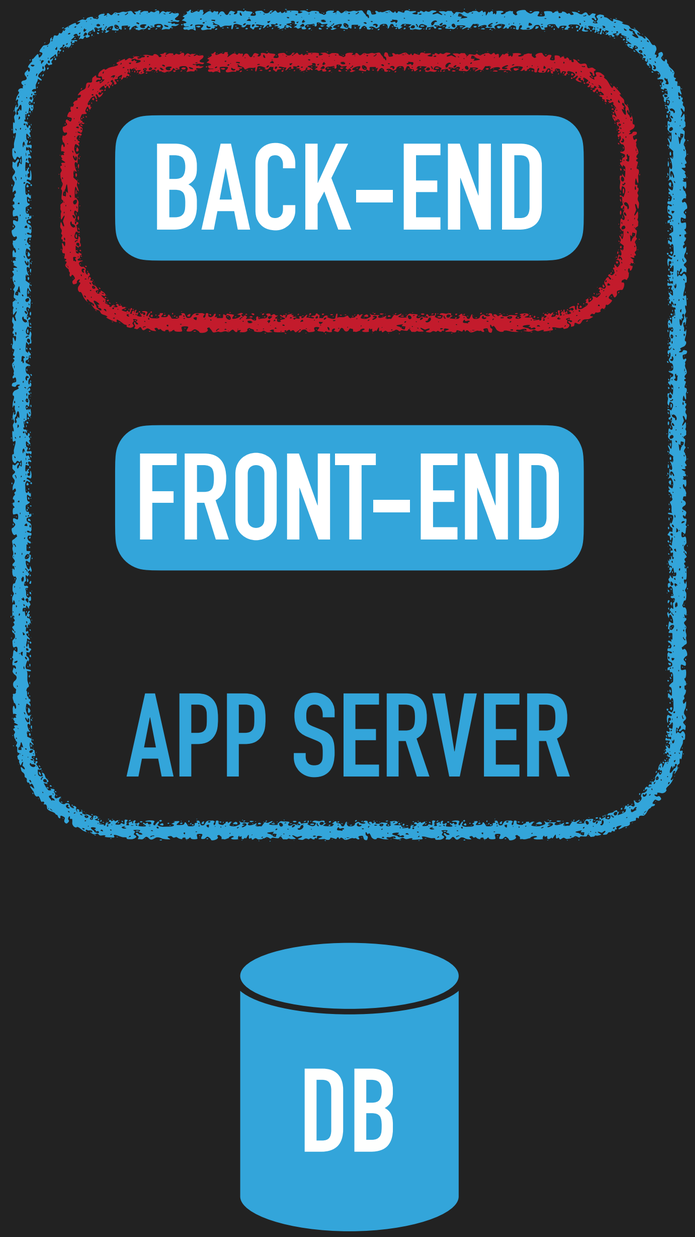
HOW DOES JHIPSTER HELP?

PICK MORE TECHNOLOGIES

BEFORE


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

CHANGES



QA

OPS

SUPPORT

MICROSERVICES & CONTAINER

BACK-END

FRONT-END

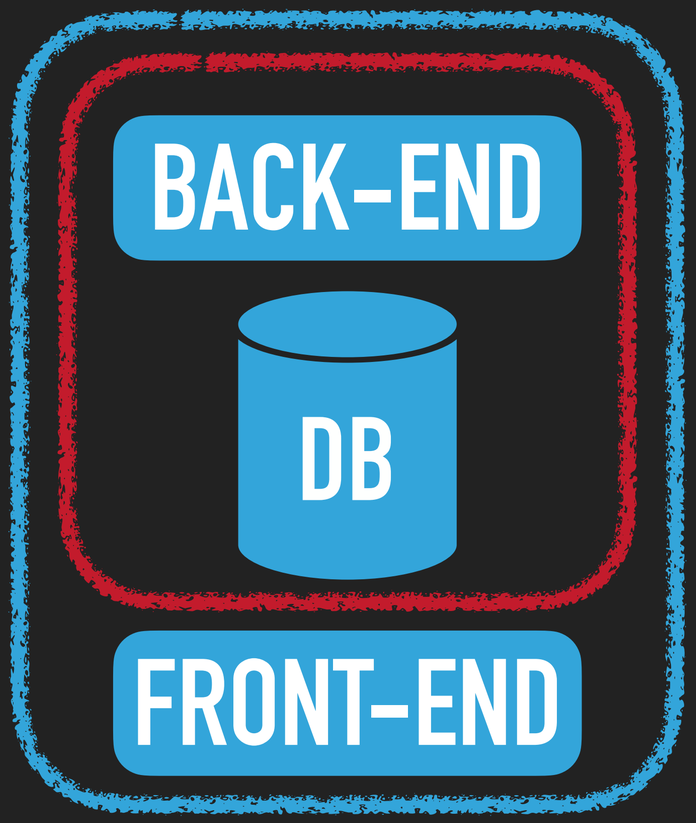
QA

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

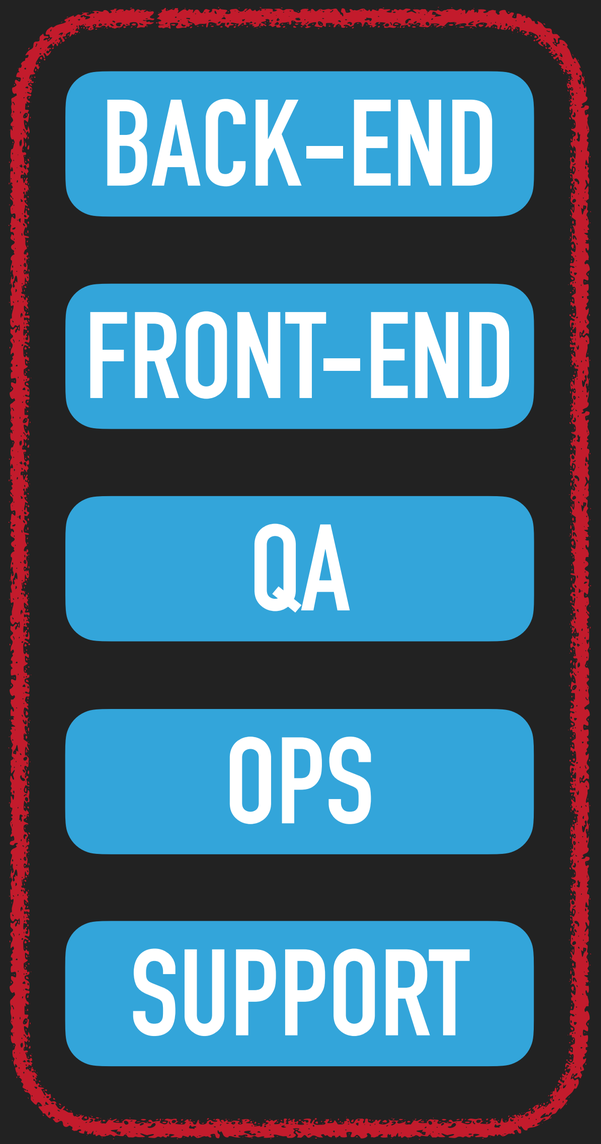
FRONT-END

QA

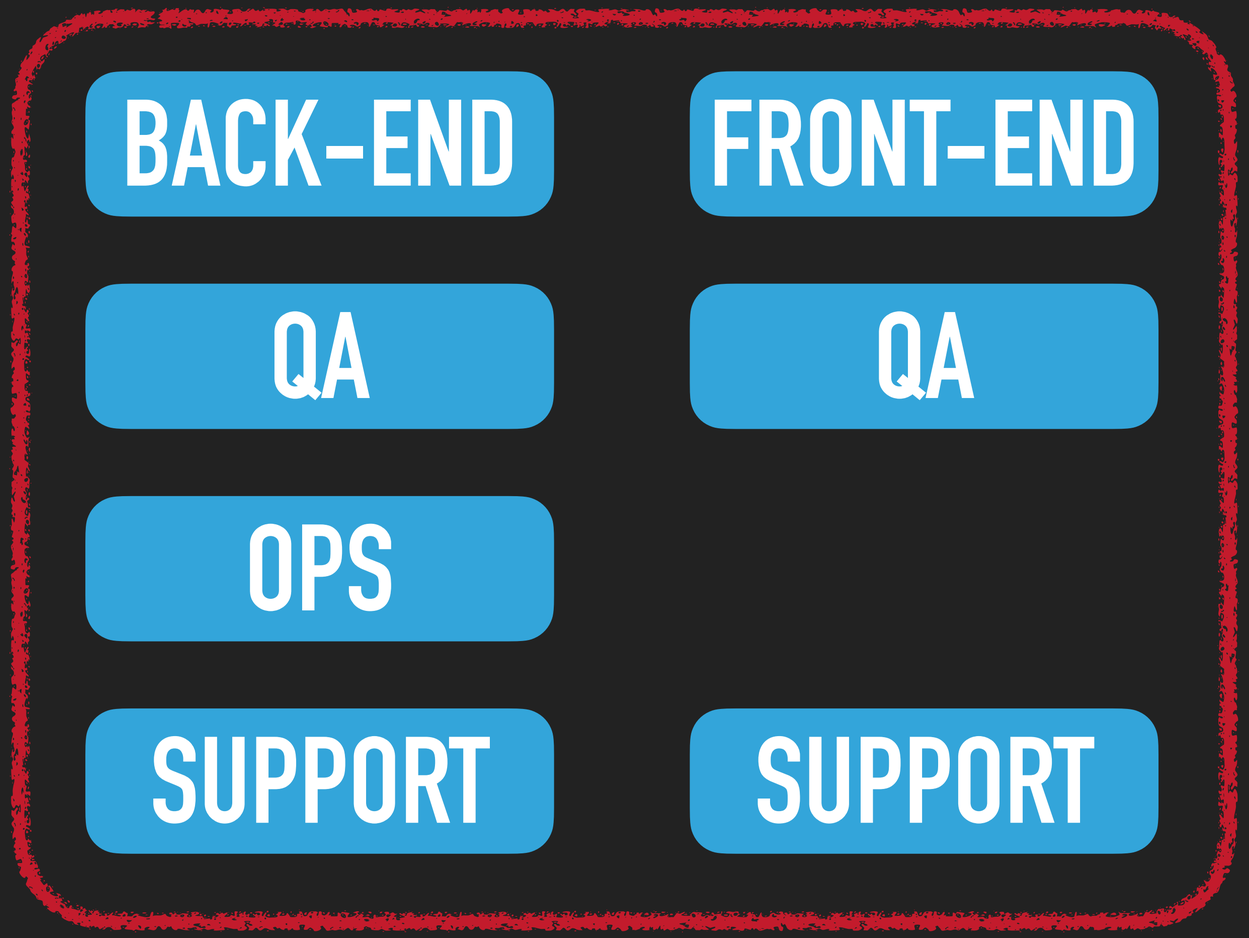
OPS

SUPPORT

DEVOPS



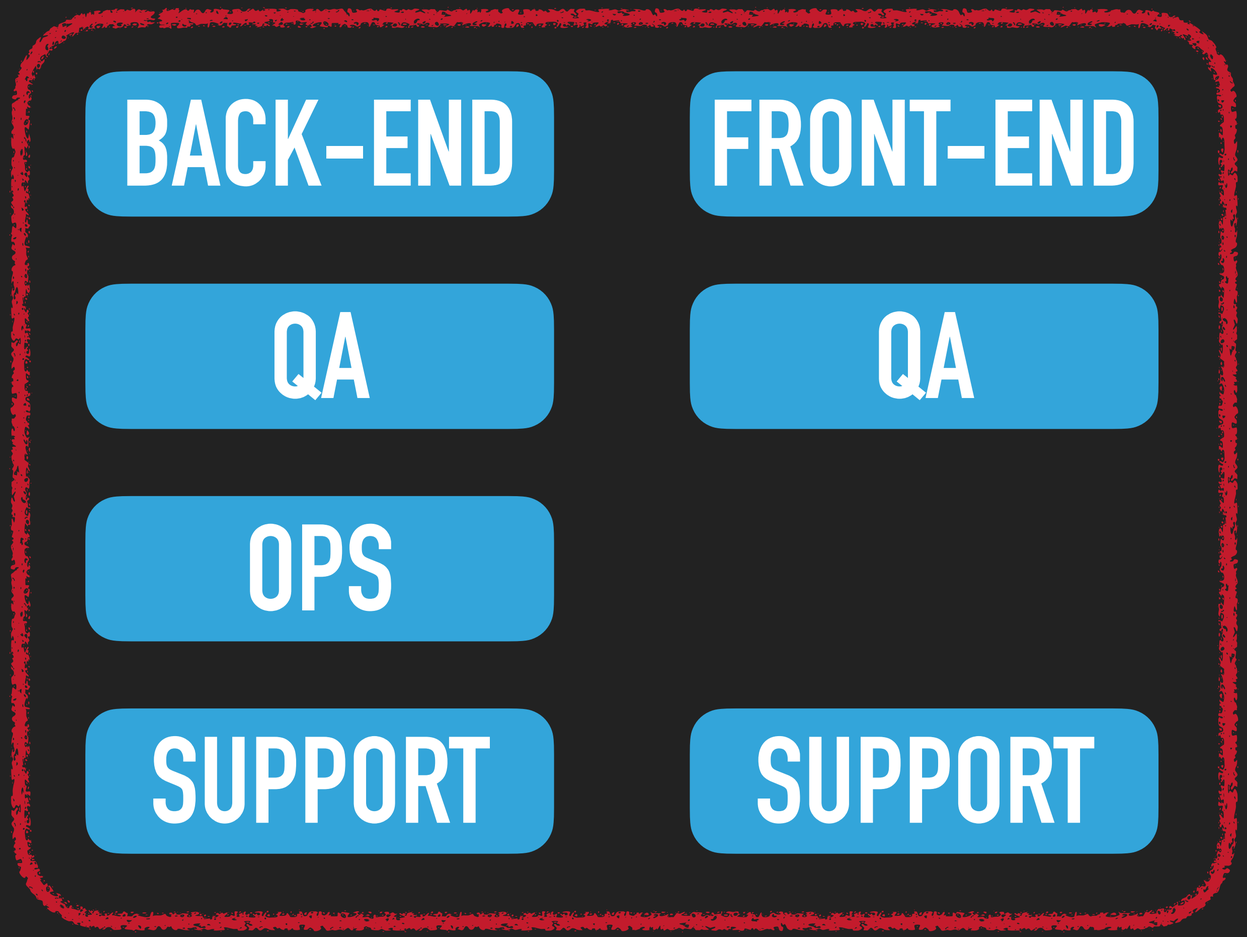
SINGLE PAGE APPLICATIONS



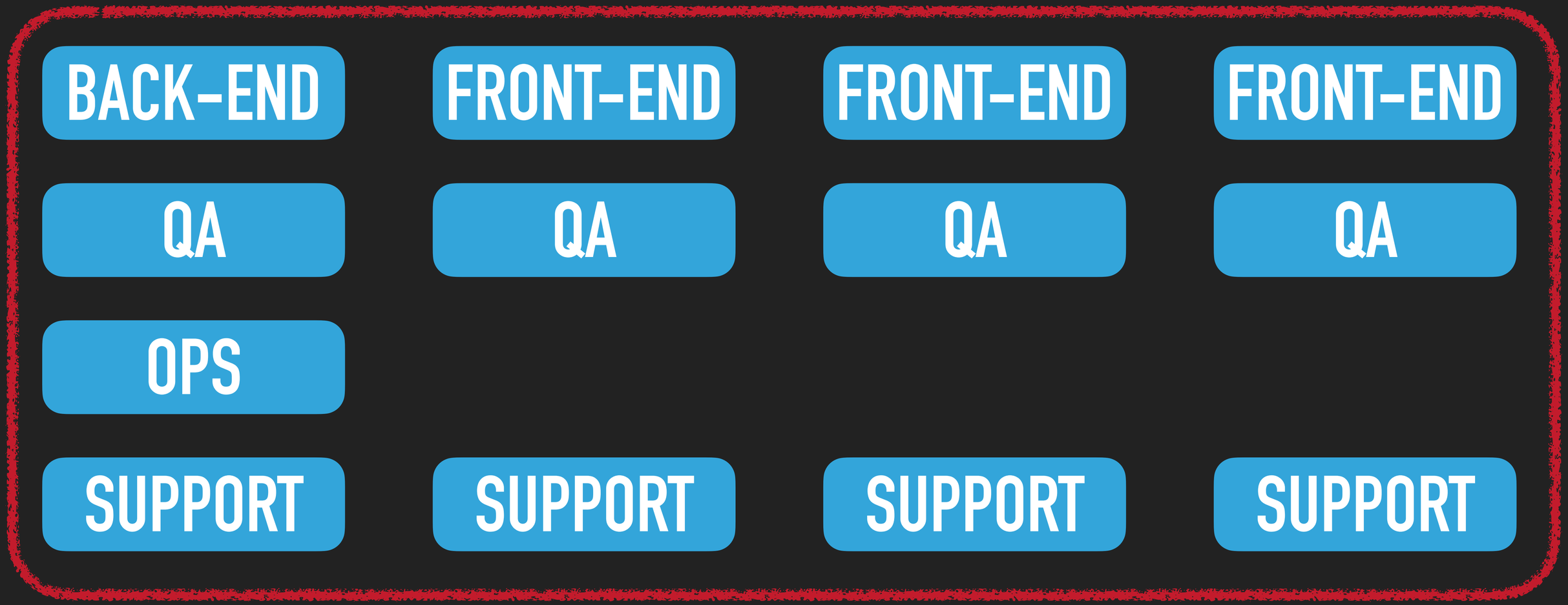
JAVA

JS

SINGLE PAGE APPLICATIONS

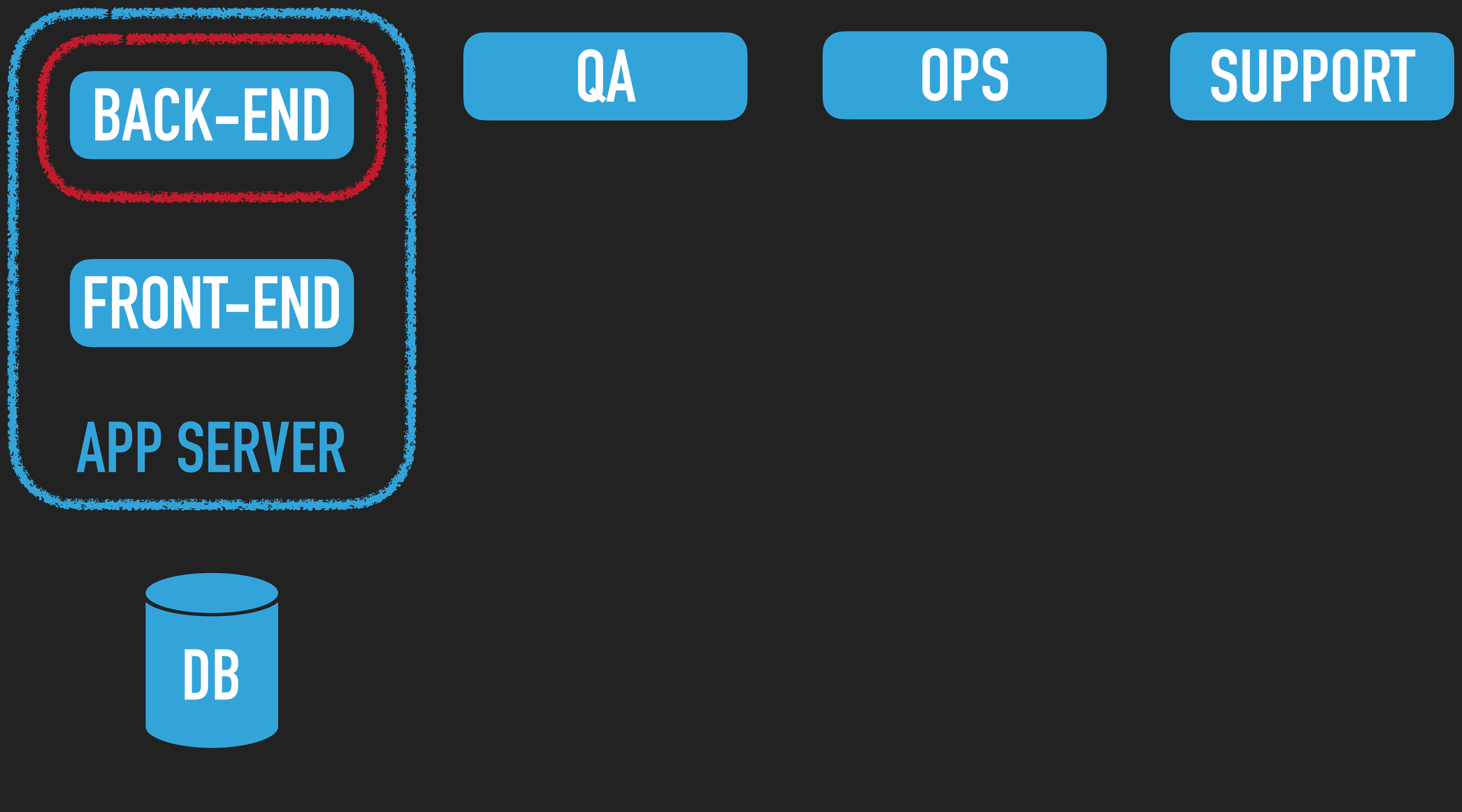


JAVA JS MOBILE APPS

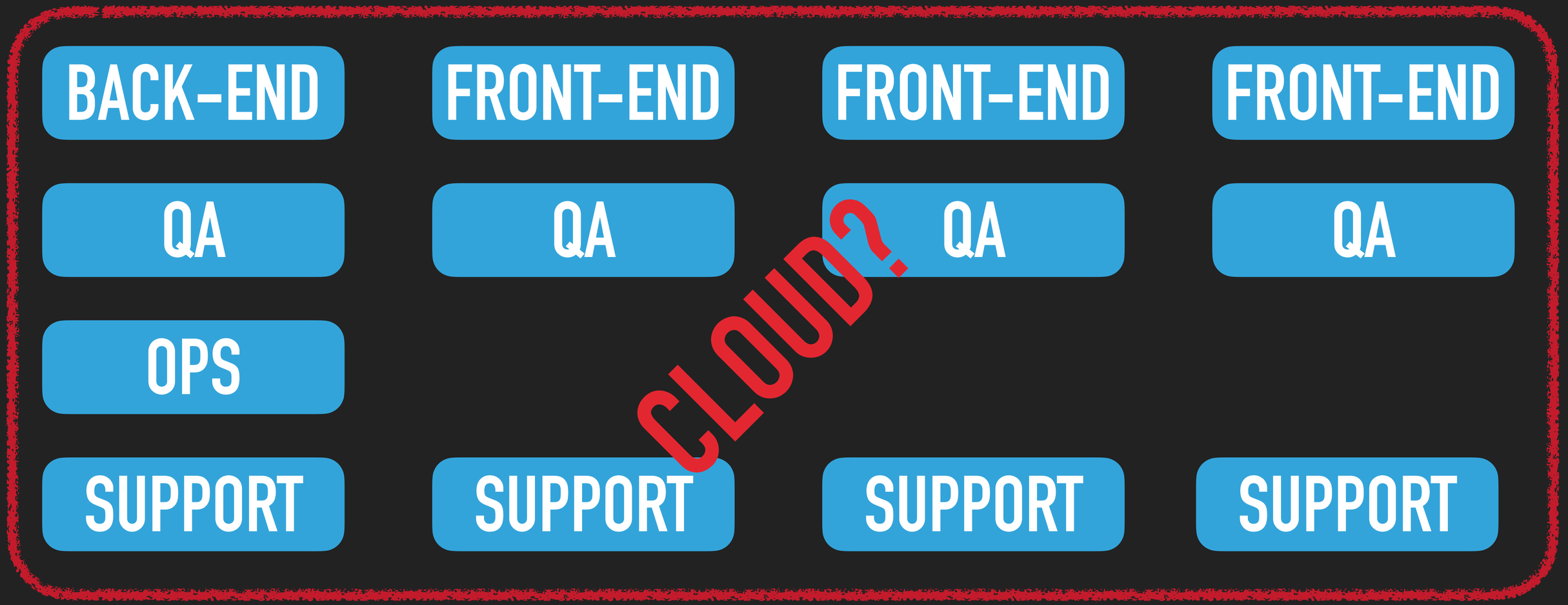

BACK-END
FRONT-END
FRONT-END
FRONT-END
QA
QA
QA
QA
OPS
SUPPORT
SUPPORT
SUPPORT
SUPPORT
JAVA
JS
IOS
ANDR.

AND WHERE DO WE USE THE CLOUD?

BEFORE


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

NOW



JAVA JS IOS ANDR.

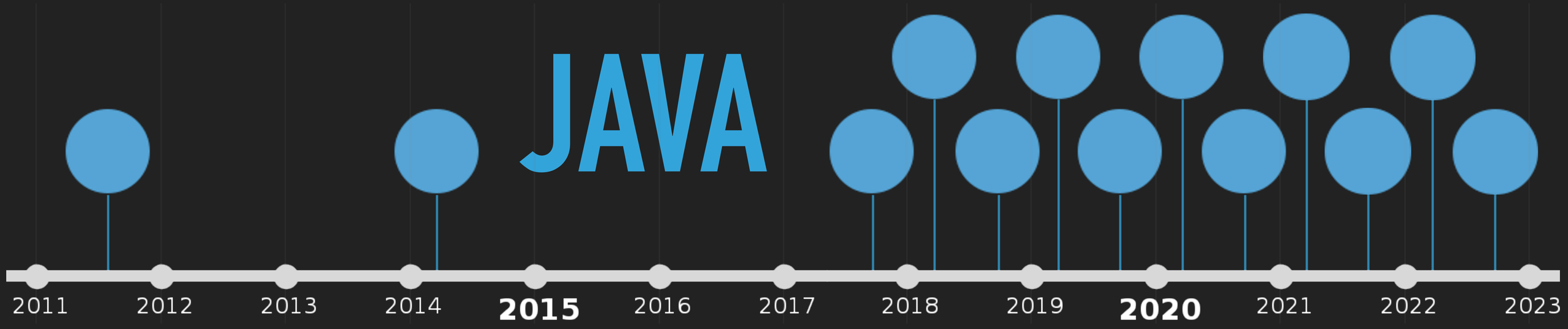
OUR PROJECTS: SOMEWHERE BETWEEN "BEFORE" AND "NOW"

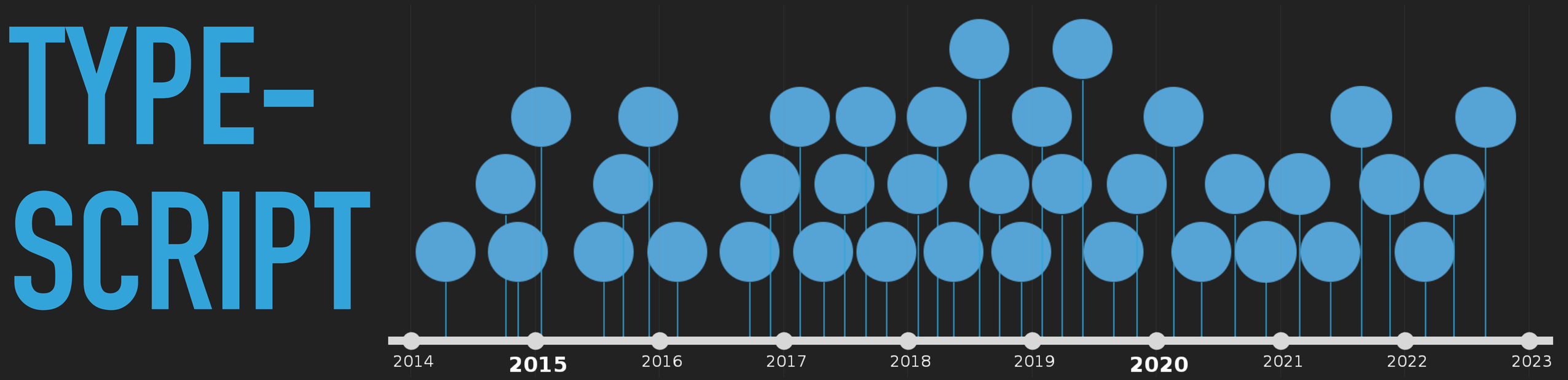
MY PROJECT: CLOSE TO "NOW" (1 FLUTTER APP INSTEAD OF 2 IOS & ANDROID ONES)

MORE TECHNOLOGIES

PICK TECHNOLOGIES MORE OFTEN

MORE RELEASES

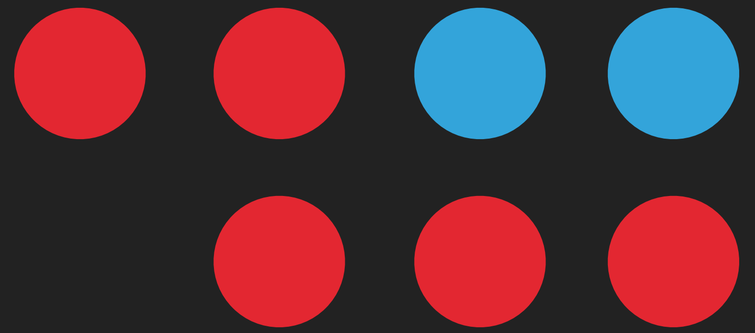

JAVA
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023


TYPE-
SCRIPT
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023

MORE ALTERNATIVES IN JAVA

FRAMEWORKS



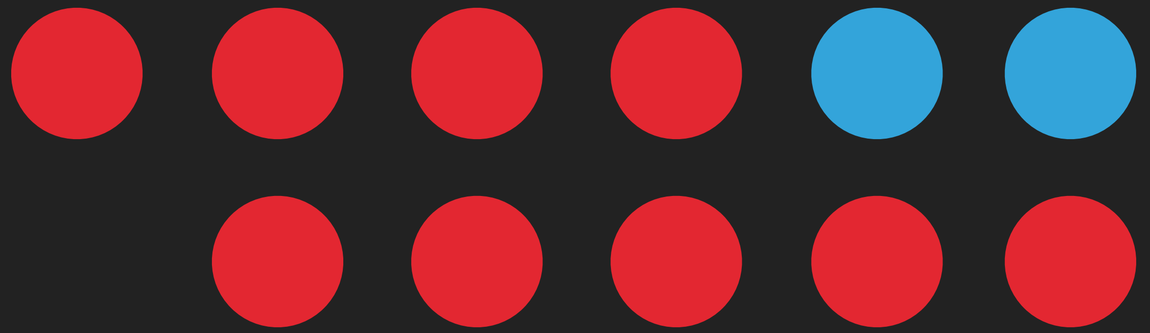


BEFORE

NOW

JAVA DISTRIBUTIONS





BEFORE

NOW

PICK MORE TECHNOLOGIES
MORE OFTEN

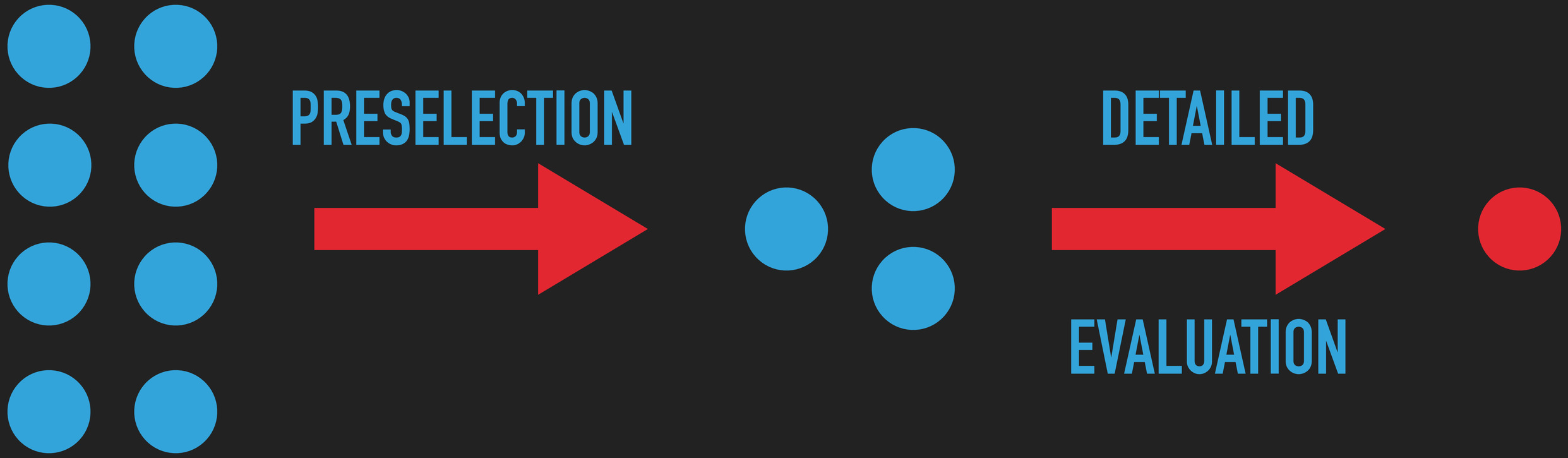
~~JAVA: PICK MORE TECHNOLOGIES MORE OFTEN~~

HOW DO WE PICK TECHNOLOGIES?

WHAT IS JHIPSTER?

HOW DOES JHIPSTER HELP?

HOW DO WE PICK TECHNOLOGIES?


PRESELECTION
DETAILED
EVALUATION

ELIMINATE CANDIDATES

WORKS?

SCALES?

POPULAR?

EXTENSIBLE?

SECURE?

RISK?

COMPLIANCE?

PRESELECTION

UNKNOWN TECHNOLOGIES!

WHICH CANDIDATES?

HOW TO SCORE?

HOW TO ELIMINATE?

PRESELECTION: PROBLEMS

OUR ENVIRONMENT

OUR DATA

OUR IDE

BUILD & DEPLOY

DEBUG

TEST

CI/CD

MONITORING

DETAILED EVALUATION

UNKNOWN TECHNOLOGIES!

HOW TO WRITE APPLICATIONS?

HOW TO USE OUR DATA?

DET. EVALUATION: PROBLEMS

PRESELECTION

DETAILED EVALUATION

THAT'S HOW WE PICK TECHNOLOGIES!

~~JAVA: PICK MORE TECHNOLOGIES MORE OFTEN~~

~~HOW~~ DO WE PICK TECHNOLOGIES?

WHAT IS JHIPSTER?

HOW DOES JHIPSTER HELP?

WHAT IS JHIPSTER?



IS A SUPERSTAR PROGRAMMER

KNOWS EVERYTHING

CONFIGURES EVERYTHING

WRITES A LOT OF CODE

JHIPSTER..



OPEN-SOURCE CODE GENERATOR

JAVA, NODE.JS, GIT, DOCKER

NO PROPRIETARY LAYER

BEST-PRACTICES CODE

JHIPSTER

I DON'T PROPOSE USING GENERATED CODE IN PRODUCTION...

. . . BUT USING GENERATED CODE FOR THROW-AWAY PROTOTYPES!

LIVE DEMO

APPLICATION GENERATION

JHIPSTER WIZARD



STARTED APPLICATION GENERATION

JHIPSTER GENERATES THE SAME APPLICATION WITH DIFFERENT TECHNOLOGIES

AROUND FOR 9 YEARS

REGULAR RELEASES (EXCEPT FOR 2020)

120K DOWNLOADS PER MONTH

MULTIPLE CORE CONTRIBUTORS, ADVISORY BOARD

SAFE PROJECT?

TEXT FILE FOR CONFIGURATION
OF APPLICATION & 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
}
```

DATA TYPES

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

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

CONFIGURATION

JHIPSTER GENERATES THE SAME APPLICATION WITH DIFFERENT TECHNOLOGIES AND OUR DATA

OPEN-SOURCE CODE GENERATOR

LIKE SUPERSTAR PROGRAMMER

NO PROPRIETARY LAYER, BUT BEST-PRACTICES CODE

THAT IS JHIPSTER

~~JAVA: PICK MORE TECHNOLOGIES MORE OFTEN~~

~~HOW~~ DO WE PICK TECHNOLOGIES?

~~WHAT IS JHIPSTER~~?

HOW DOES JHIPSTER HELP?

HOW DOES JHIPSTER HELP?

UNKNOWN TECHNOLOGIES!

WHICH CANDIDATES?

HOW TO SCORE?

HOW TO ELIMINATE?

PRESELECTION: PROBLEMS

UNKNOWN TECHNOLOGIES!

HOW TO WRITE APPLICATIONS?

HOW TO USE OUR DATA?

DET. EVALUATION: PROBLEMS

PRESELECTION: BY JHIPSTER

DETAILED EVALUATION: JHIPSTER GENERATES COMPLETE APPLICATION, LEARN FROM UNKNOWN CODE!

HOW DOES JHIPSTER HELP?

DATABASES: ALL RELATIONAL EXCEPT FOR DB2;
NOSQL: MONGODB, CASSANDRA, COUCHBASE, NEO4J

FRAMEWORKS: SPRING BOOT, QUARKUS, MICRONAUT,
NODE.JS, .NET (PARTIALLY BETA)

FRONT-END: REACT, ANGULAR, VUE.JS

PRESELECTION: JHIPSTER

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

PRESELECTION: JHIPSTER

GENERATE APPLICATION WITH ALL TECHNOLOGIES

IMPORT OUR DATA MODEL AS JDL

CHECK APPLICATION IN OUR ENVIRONMENT

DET. EVALUATION: JHIPSTER

~~OUR ENVIRONMENT~~

~~OUR DATA~~

~~OUR IDE~~

~~BUILD & DEPLOY~~

~~DEBUG~~

~~TEST~~

~~CI/CD~~

~~MONITORING~~

DET EVALUATION: JHIPSTER

LIVE DEMO

INSPECT APPLICATION



INSPECT APPLICATION

WE DON'T HAVE TO
UNDERSTAND THE
JHIPSTER CODE!

LEARN FROM UNKNOWN CODE?

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;
```

DIFFERENT TYPE

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

SIMILAR!

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

PRESENTATION CODE IN SEPARATE 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, PROPRIETARY ATTRIBUTES

```
<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 - ALL IN JAVASCRIPT FILE

```
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>
```

NO 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

WE CAN LEARN FROM UNKNOWN CODE!

TECHNOLOGY NOT IN JHIPSTER

HOW PRODUCTIVE WITH TECHNOLOGIES?

WHEN DOES JHIPSTER NOT HELP?

PRESELECTION: BY JHIPSTER

DETAILED EVALUATION: JHIPSTER GENERATES COMPLETE APPLICATION, LEARN FROM UNKNOWN CODE!

HOW DOES JHIPSTER HELP?

JAVA: PICK MORE TECHNOLOGIES MORE OFTEN

HOW DO WE PICK TECHNOLOGIES?

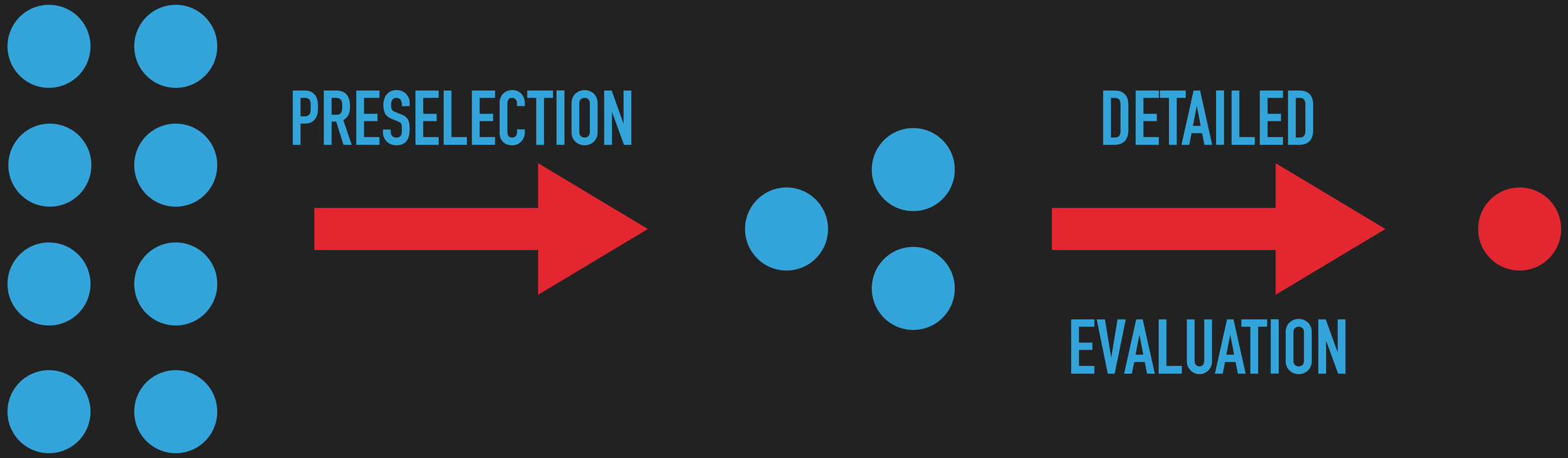
WHAT IS JHIPSTER?

HOW DOES JHIPSTER HELP?

SUMMARY

WE DEVELOPERS PICK
MORE TECHNOLOGIES
MORE OFTEN

HOW DO WE PICK TECHNOLOGIES?


PRESELECTION
DETAILED
EVALUATION

JHIPSTER DOES
PRESELECTION AND HELPS
WITH DETAILED EVALUATION

OPEN-SOURCE CODE GENERATOR

LIKE SUPERSTAR PROGRAMMER

NO PROPRIETARY LAYER, BUT BEST-PRACTICES CODE

THAT IS JHIPSTER

JHIPSTER GENERATES THE SAME APPLICATION WITH DIFFERENT TECHNOLOGIES AND OUR DATA

TECHNOLOGY NOT IN JHIPSTER

HOW PRODUCTIVE WITH TECHNOLOGIES?

WHEN DOES JHIPSTER NOT HELP?

PICK TECHNOLOGIES & TOOLS FASTER WITH JHIPSTER!


The End

MY TALK DOESN'T STOP WHEN I STOP TALKING!™



SLIDES

JHIPSTER TUTORIAL

ADDITIONAL INFOS

FEEDBACK & NEWSLETTER

BPF.LI/JET