WILD CODE SCHOOL

Little Helpers

Useful Tools for (Java) Developers

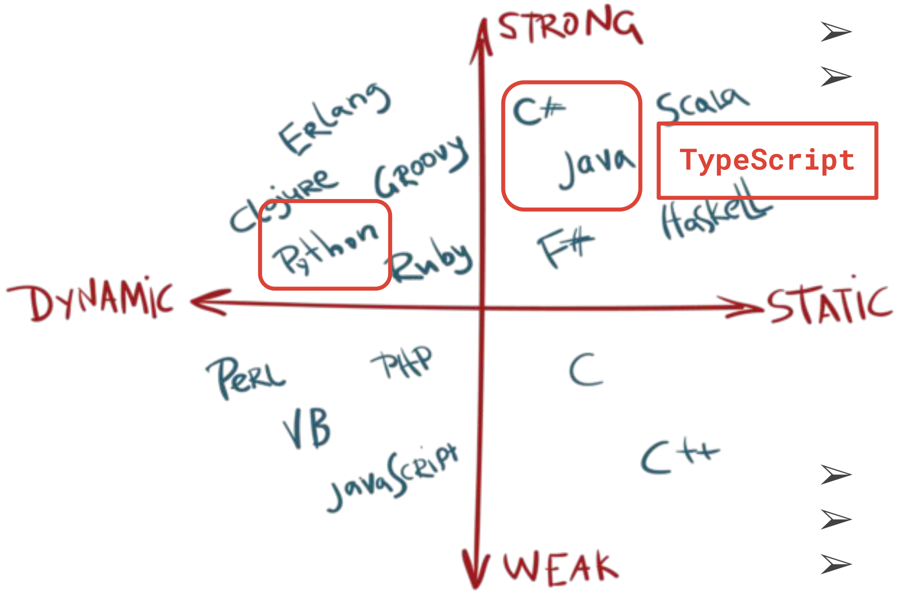
WILD
CODE
SCHOOL

- Why do some tech stacks need tools?
- Fighting Verbosity: Lombok
- Better Tooling: Google Guava
- Java Swiss Army Knife: Apache Commons
- Back to Excel: Apache POI

- ❖ (Aspect-Oriented Programming: AspectJ / Spring)
- ❖ (Rapid Application Development: JHipster)
- ❖ (Spring with more Fun: Spring Boot Dev Tools)
- ❖ (Better Testing: Mockito, AssertJ & ArchUnit)

Why does Java need Tools and Python not?

Static vs Dynamic and Weak vs Strong

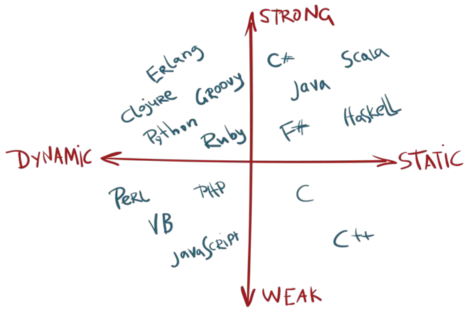


- **strong** and **weak typing** refers to **type consistency**
- **static** and **dynamic typing** refers to **the time** when names are **bound to types**

- **C#**, **Java** and **Typescript** are **static** & **strong**
- **Python**, **Ruby** and **Erlang** are **dynamic** & **strong**
- **JavaScript** and **Perl** are **weak** & **dynamic**

taken from https://medium.com/android-news/magic-lies-here-statically-typed-vs-dynamically-typed-languages-d151c7f95e2b

Static vs Dynamic and Weak vs Strong



Python: Dynamic Strong Typing

```
value = 21;                  # type assigned as int at runtime.
value = value + "dot";       # type-error, string and int cannot be concatenated.
print(value);
```

JavaScript: Dynamic Weak Typing

```
value = 21;
value = value + "dot";      # “21dot”
console.log(value);         # no error, implicit conversion between unrelated types
```

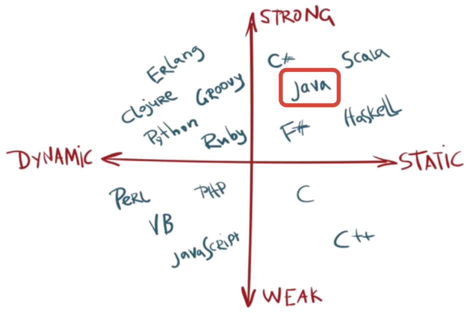
Java: Static Strong Typing

```
var value = 21;              // type int set at compile time (type inference)
value = value + "dot";       // compile-time-error
System.out.println(value);
```

TypeScript: Static (Pre-Compile) Strong Typing

```
var value = 21;             # type assigned as int at compile-time.
value = value + "dot";      # (pre-)compile-time-error
console.log(value);
```

Static vs Dynamic and Weak vs Strong

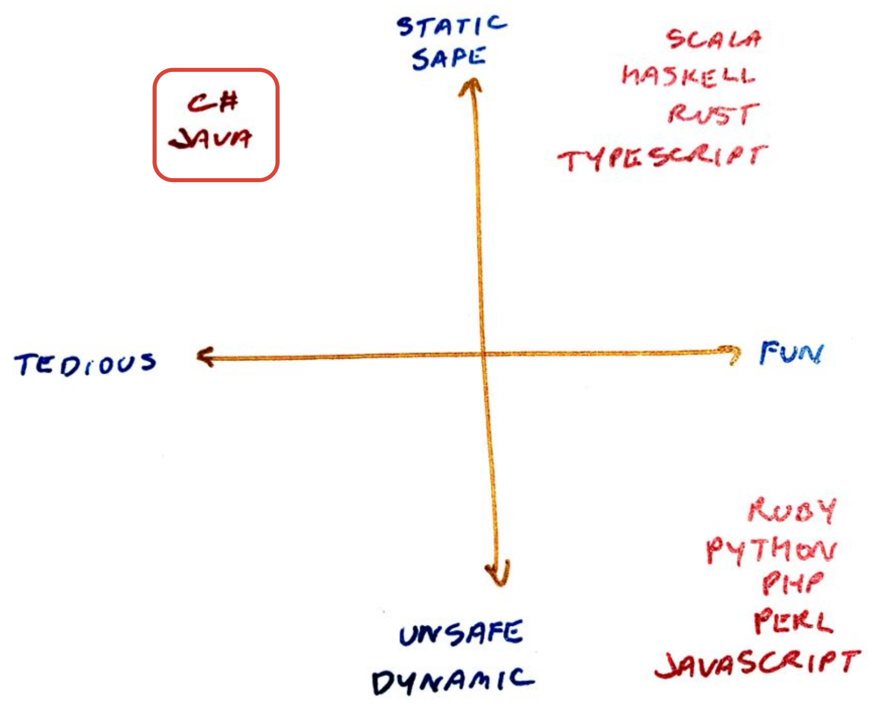


Java is a **strongly** and **statically typed** language and favors **simplicity of syntax**.

With these characteristics, Java tends to be **very verbose**, requiring a lot of **boilerplate code** and **tedious typing**.

Mitigation of these shortcomings is typically done by using **highly dynamic frameworks** (like Spring Framework), favour **Convention-over-Configuration** and heavily rely on **code generation frameworks** (like Lombok).

Fun vs Tedious vs Static vs Dynamic



How can we bring FUN back to Java?

- ❖ **Avoid boilerplate** code, **focus** on **business logic**
- ❖ **Provide useful libraries** for missing functionality
- ❖ **Copy patterns** from other languages / tech stacks
- ❖ **Extend** existing **codebases**
- ❖ **Provide** more **choices** for developers

taken from https://medium.com/@markcanlasnyc/scala-saturday-static-vs-dynamic-typing-2bbe6a283233

WILD CODE SCHOOL

Fighting Verbosity: Lombok

Ease Java Developent & Focus on Logic

```java
public class NoLombokPojo {

   private String attribute;
   private LocalDate created;
   private BigDecimal value;

   public NoLombokPojo(String attribute, LocalDate created, BigDecimal
value) {
       this.attribute = attribute;
       this.created = created;
       this.value = value;
   }
   public String getAttribute() {
       return attribute;
   }
   public void setAttribute(String attribute) {
       this.attribute = attribute;
   }
   public LocalDate getCreated() {
       return created;
   }
   public void setCreated(LocalDate created) {
       this.created = created;
   }
   public BigDecimal getValue() {
       return value;
   }
   public void setValue(BigDecimal value) {
       this.value = value;
   }
   @Override
   public boolean equals(Object o) {
       if (this == o) return true;
       if (o == null || getClass() != o.getClass()) return false;
       NoLombokPojo that = (NoLombokPojo) o;
       return attribute.equals(that.attribute) &&
created.equals(that.created) && value.equals(that.value);
   }

}
```

lombok

- ➢ Avoid repetitive code
- ➢ Less error-prone code
- ➢ Consistent behaviour over projects
- ➢ Best practices baked in

```java
@Data
public class LombokPojo {

   private String attribute;
   private LocalDate created;
   private BigDecimal value;

}
```

`@Data` is the short form for

`@RequiredArgsConstructor, @ToString, @EqualsAndHashCode, @Getter, @Setter`

Constructors: Automatic Constructor Creation

```java
public class LombokConstructorPojo {

   private @NonNull String id;
   private LocalDate created;
   private BigDecimal value;

   public LombokConstructorPojo(@NonNull String id) {
       this.id = id;
   }

   public LombokConstructorPojo(
           @NonNull String id,
           LocalDate created,
           BigDecimal value) {
       this.id = id;
       this.created = created;
       this.value = value;
   }

   public LombokConstructorPojo() {
   }

}
```

➢

lombok

```java
@NoArgsConstructor
@RequiredArgsConstructor
@AllArgsConstructor
public class LombokPojo {

    private String attribute;
    private LocalDate created;
    private BigDecimal value;

}
```

@Value: Immutable Classes made easy

```
public final class LombokConstructorPojo {

   private final @NonNull String id;
   private final LocalDate created;
   private final BigDecimal value;

   public LombokConstructorPojo(
           @NonNull String id,
           LocalDate created,
           BigDecimal value) {
       this.id = id;
       this.created = created;
       this.value = value;
   }

   public @NonNull String getId() {
       return this.id;
   }

   public LocalDate getCreated() {
       return this.created;
   } [...]

}
```

lombok

- `@Value`

```
@Value
public class LombokConstructorPojo {

   private @NonNull String id;
   private LocalDate created;
   private BigDecimal value;

}
```

@Builder: Immutable Classes made easy

```
public class LombokConstructorPojo {

   private @NonNull String id;
   private LocalDate created;
   private BigDecimal value;

   public static LombokConstructorPojoBuilder builder() {
      return new LombokConstructorPojoBuilder();
   }

   public static class LombokConstructorPojoBuilder {
       private @NonNull String id;
       private LocalDate created;
       private BigDecimal value;

       public LombokConstructorPojoBuilder id(@NonNull String id) {
           this.id = id;
           return this;
       }

       public LombokConstructorPojoBuilder created(LocalDate creatd){
           this.created = creatd;
           return this;
       }

       public LombokConstructorPojo build() {
           return new LombokConstructorPojo(id, created, value);
       }
   }

}
```

`@Builder`

lombok

```
@Builder
public class LombokConstructorPojo {

   private @NonNull String id;
   private LocalDate created;
   private BigDecimal value;

}
```

Automatic Printer: @ToString

```java
public class NoLombokPojo {

   private String attribute;
   private LocalDate created;
   private BigDecimal value;

   public NoLombokPojo(String attribute, LocalDate created, BigDecimal
value) {
       this.attribute = attribute;
       this.created = created;
       this.value = value;
   }
   public String getAttribute() {
       return attribute;
   }
   public void setAttribute(String attribute) {
       this.attribute = attribute;
   }
   public LocalDate getCreated() {
       return created;
   }
   public void setCreated(LocalDate created) {
       this.created = created;
   }
   public BigDecimal getValue() {
       return value;
   }
   public void setValue(BigDecimal value) {
       this.value = value;
   }
   @Override
   public boolean equals(Object o) {
       if (this == o) return true;
       if (o == null || getClass() != o.getClass()) return false;
       NoLombokPojo that = (NoLombokPojo) o;
       return attribute.equals(that.attribute) &&
created.equals(that.created) && value.equals(that.value);
   }

}
```

- `@ToStrin`

```java
@Data
public class LombokPojo {

   private String attribute;
   private LocalDate created;
   private BigDecimal value;

}
```

Are we the same? @EqualsAndHashCode

```java
public class NoLombokPojo {

   private String attribute;
   private LocalDate created;
   private BigDecimal value;

   public NoLombokPojo(String attribute, LocalDate created, BigDecimal
value) {
       this.attribute = attribute;
       this.created = created;
       this.value = value;
   }
   public String getAttribute() {
       return attribute;
   }
   public void setAttribute(String attribute) {
       this.attribute = attribute;
   }
   public LocalDate getCreated() {
       return created;
   }
   public void setCreated(LocalDate created) {
       this.created = created;
   }
   public BigDecimal getValue() {
       return value;
   }
   public void setValue(BigDecimal value) {
       this.value = value;
   }
   @Override
   public boolean equals(Object o) {
       if (this == o) return true;
       if (o == null || getClass() != o.getClass()) return false;
       NoLombokPojo that = (NoLombokPojo) o;
       return attribute.equals(that.attribute) &&
created.equals(that.created) && value.equals(that.value);
   }

}
```

➢ `@EqualsAndHashCode`

lombok

```java
@Data
public class LombokPojo {

   private String attribute;
   private LocalDate created;
   private BigDecimal value;

}
```

Pluggable Logging with SLF4J

- ➢ Avoid repetitive code
- ➢ Less error-prone code
- ➢ Consistent behaviour over projects
- ➢ Best Practices baked in

```
public class LombokLog {

   private static final Logger log =
    org.slf4j.LoggerFactory.getLogger(LombokConstructorPojo.class);

[...]
```

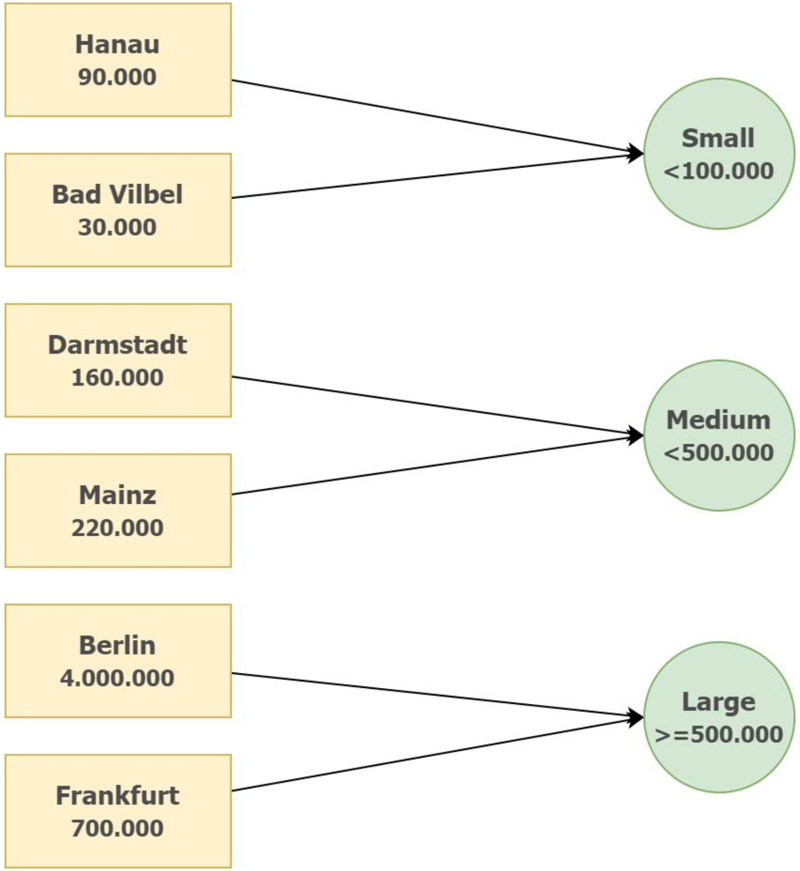
lombok

```
@Slf4j
public class LombokLog {

[...]
log.info("Hello from Logger");

}
```

WILD
CODE
SCHOOL

Google Core Libraries for Java: Guava

City Lights - Sample Collection



```
@Data
public class City {

    private @NonNull String name;
    private @NonNull Integer population;
    private LocalDate founded;
    private boolean capital;

}
```

WILD CODE SCHOOL

Java Swiss Army Knife: Apache Commons

Reason for Payment - Sample Texts

"Thanks for the nice Lunch."
"Max Muster Invoice No 551662781"
"Remittance Item #188655"
"September 2022 Loan Repayment Max Mustermann"
"Invoice Reference No 8988826/2022/145"

Abbreviate the text to max 20 characters.

Please make sure that no words are cut.

“Thanks for the nice…”
“Max Muster Invoice…”
“Remittance Item #188…”
“September 2022 Loan…”
“Invoice Reference…”

“Thanks for the ni…”
“Max Muster Invoic…”
“Remittance Item #…”
“September 2022 Lo…”
“Invoice Reference…”

WILD CODE SCHOOL

The Java API for Microsoft Documents: Apache POI

Links and other information

Project Lombok

- ❖ Project Lombok: https://projectlombok.org/features/all
- ❖ Baeldung on Lombok: https://www.baeldung.com/intro-to-project-lombok

Google Guava

- ❖ Baeldung on Guava Maps: https://www.baeldung.com/guava-maps
- ❖ Baeldung on Guava StringUtils: https://www.baeldung.com/guava-string-charmatcher
- ❖ Java Dev Central on Guava: https://javadevcentral.com/category/google-guava
- ❖ Guava Documentation: https://github.com/google/guava/wiki/

Apache Commons

- ❖ Apache Commons (Proper): https://commons.apache.org/
- ❖ Java Dev Central on Commons: https://javadevcentral.com/category/apache-commons
- ❖ Java Dev Central on Commons Text: https://javadevcentral.com/apache-commons-text-wordutils
- ❖ Java Dev Central on Commons Lang: https://javadevcentral.com/apache-commons-lang-randomstringutils