Constructors

Different constructor syntax

Concise primary constructor

```
class Person(val name: String, val age: Int)
```

Full primary constructor syntax (1)

constructor parameter

```
class Person(name: String) {

    init {

    }
}
```

constructor body

Full primary constructor syntax (2)

constructor parameter

```
class Person(name: String) {

    val name: String
```

constructor body

```
    init {
        this.name = name
    }
}
```

`val/var` on a parameter creates a property

```
class Person(name: String) {

    val name: String

    init {
        this.name = name
    }
}
```

=

```
class Person(val name: String)
```

Changing visibility of a constructor

```
class InternalComponent
internal constructor(name: String) {
    …
}
```

Secondary constructor

primary constructor

```
class Rectangle(val height: Int, val width: Int) {
```

secondary constructor

```
    constructor(side: Int) : this(side, side) { … }
}
```

Secondary constructor

```
class Rectangle(val height: Int, val width: Int) {

    constructor(side: Int) : this(side, side) { … }
}
```

`this(...)` calls another constructor of the same class

Different syntax for inheritance

The same syntax for `extends` & `implements`

```
interface Base
class BaseImpl : Base

open class Parent
class Child : Parent()
```

The same syntax for `extends` & `implements`

```
interface Base
class BaseImpl : Base

open class Parent
class Child : Parent()
```

constructor call

Calling a constructor of the parent class

```
open class Parent(val name: String)
class Child(name: String) : Parent(name)

open class Parent(val name: String)
class Child : Parent {
    constructor(name: String, param: Int) : super(name)
}
```

Initialization order. What will be printed?

```
open class Parent {
    init { print("parent ") }
}

class Child : Parent() {
    init { print("child ") }
}

fun main(args: Array<String>) {
    Child()
}
```

1. `child`
2. `child parent`
3. `parent child`

?

Initialization order. What will be printed?

```
open class Parent {
    init { print("parent ") }
}

class Child : Parent() {
    init { print("child ") }
}

fun main(args: Array<String>) {
    Child()
}
```

1. `child`
2. `child parent`
3. `parent child`