Lambda with receiver

Extension Function & Lambda

$\downarrow$

Lambda with receiver

The `with` function

```kotlin
val sb = StringBuilder()
sb.appendln("Alphabet: ")
for (c in 'a'..'z') {
    sb.append(c)
}
sb.toString()
```

with is a function

```kotlin
val sb = StringBuilder()
with (sb) {
    appendln("Alphabet: ")
    for (c in 'a'..'z') {
        append(c)
    }
    toString()
}
```

Lambda with receiver

`with` is a function

```
val sb = StringBuilder()
with (sb, { ->
    this.appendln("Alphabet: ")
    for (c in 'a'..'z') {
        this.append(c)
    }
    this.toString()
})
```

lambda is its second argument

`this` is an implicit receiver in the lambda

`this` can be omitted

```
val sb = StringBuilder()
with (sb) {
    appendln("Alphabet: ")
    for (c in 'a'..'z') {
        append(c)
    }
    toString()
}
```

Lambda with receiver

lambda with
implicit `this`

```
val sb = StringBuilder()
with (sb) {
    appendln("Alphabet: ")
    for (c in 'a'..'z') {
        this.append(c)
    }
}
```

What is the type of `this`?

```
val sb = StringBuilder()
with (sb) {
    this.appendln("Alphabet: ")
    for (c in 'a'..'z') {
        this.append(c)
    }
}
```

?

What is the type of `this`?

```
val sb = StringBuilder()
with (sb) {
    this.appendln("Alphabet: ")
    for (c in 'a'..'z') {
        this.append(c)
    }
}
```

StringBuilder

Extension function vs lambda with receiver

regular function	regular lambda
extension function	lambda with receiver

Extension function vs lambda with receiver

regular function	regular lambda
extension function	lambda with receiver

```
fun String.lastChar() =
    this.get(this.length - 1)
```

```
buildString {
    this.append("...")
}
```

Lambda vs lambda with receiver

regular function	regular lambda
extension function	lambda with receiver

```
val isEven: (Int) -> Boolean = { it % 2 == 0 }
val isOdd: Int.() -> Boolean = { this % 2 == 1 }
```

```
isEven(0)
1.isOdd()
```

calling as regular function

calling as extension function

Another example: `buildString`

lambda with receiver

```
val s: String = buildString {
    appendln("Alphabet: ")
    for (c in 'a'..'z') {
        append(c)
    }
}
```

The buildString function

Creates a StringBuilder

Calls the specified actions on a stringBuilder

Returns String as a result

```
inline fun buildString(
        builderAction: StringBuilder.() -> Unit
): String {
    val stringBuilder = StringBuilder()
    stringBuilder.builderAction()
    return stringBuilder.toString()
}
```

```
buildString {
    this.append("...")
}
```

Complete `buildString` definition

Creates a `StringBuilder`

Calls the specified actions on a `stringBuilder`

Returns `String` as a result

```
inline fun buildString(
        builderAction: StringBuilder.() -> Unit
): String {
    val stringBuilder = StringBuilder()
    ...
    return stringBuilder.toString()
}
```

```
buildString {
    this.append("...")
}
```

?

The buildString function

Creates a StringBuilder

Calls the specified actions on a stringBuilder

Returns String as a result

```
inline fun buildString(
        builderAction: StringBuilder.() -> Unit
): String {
    val stringBuilder = StringBuilder()
    stringBuilder.builderAction()
    return stringBuilder.toString()
}
```

```
buildString {
    this.append("...")
}
```

The `with` function declaration

```kotlin
inline fun <T, R> with(
  receiver: T,
  block: T.() -> R
): R = receiver.block()
```

```kotlin
with (sb) {
    appendln("Alphabet: ")
    ...
}
```

HTML Builders

lambdas with receiver

```
html {
    table {
        for (product in products) {
            tr {
                td { text(product.description) }
                td { text(product.price) }
                td { text(product.popularity) }
            }
        }
    }
}
```

Gradle Build Script in Kotlin

```
plugins {
    application
    kotlin("jvm") version "1.1.51"
}

application {
    mainClassName = "samples.HelloWorldKt"
}

dependencies {
    compile(kotlin("stdlib"))
}

repositories {
    jcenter()
}
```

lambdas with receiver