

## R reference card: **Classes: S3, S4, R5**

### S3 & S4 classes <sup>1</sup>

#### Class representation:

We don't declare class representation for **S3** class.

#### For S4 class:

```
setClass("personS4",  
representation(name="character",age="numeric"))
```

In **S4**, while creating objects they have to be compatible with a given prototype. A trick for this:

```
setClassUnion("NumbChar", c("numeric", "character"))
```

```
setClass("person2", representation(name="character",  
wiek="NumbChar"))
```

#### Methods:

We add suffix for **S3** class:

```
print.personS3 <- function(x) {  
  cat(x$name, " is a person of age ", x$age, "\n\n")  
}
```

#### For S4 class:

```
setMethod("print", "personS4",  
function(x) {  
  cat(x@name, " is a person of age ", x@age, "\n\n")  
}))
```

#### Creating of objects from classes:

##### For S3:

```
o1 <- list(name = "Przemek", age = 27)  
class(o1) = "personS3"
```

##### For S4:

With function new():

```
o2 <- new("personS4", name="Przemek", age=27)
```

---

<sup>1</sup> Based on: <http://tofesi.mimuw.edu.pl/~cogito/ProgramRMIMUW/S3S4.R>

#### Example:

```
setClass("person", representation(name="character",  
age="numeric")
```

#New constructor for class person:

```
setMethod("initialize", "person",  
function(.Object, name = character("Empty"), age =  
numeric(0)) {  
  if (nargs() > 1) {  
    if(age > 120 || age < 0)  
      stop("Age was not correctly given")  
    .Object@name <- name  
    .Object@age <- age  
  }  
  .Object  
}))
```

### R5 class<sup>2</sup>

**Help:** ?ReferenceClasses

#### Creating a new object of reference based class:

```
Person <- setRefClass("Person")  
Person$new()
```

#### Inheritance:

```
setRefClass("Polygon")  
setRefClass("Regular")
```

# Specify parent classes

```
setRefClass("Triangle", contains = "Polygon")  
setRefClass("EquilateralTriangle",  
contains = c("Triangle", "Regular"))
```

#### Fields in R5:

```
setRefClass("Polygon", fields = c("sides"))  
setRefClass("Polygon", fields = list(sides = "numeric"))
```

---

<sup>2</sup> Based on: <https://github.com/hadley/devtools/wiki/R5>

#### Mutability (reference semantics):

```
Polygon <- setRefClass("Polygon", fields = c("sides"))  
square <- Polygon$new(sides = 4)
```

```
triangle <- square  
triangle$sides <- 3
```

```
square$sides
```

#### Methods:

**Syntax:** obj\$method(arg1, arg2, ...)

<<- (to modify fields) calls accessor functions if defined.

**R5 classes** inherit from the same superclass, envRefClass

#### Examples:

```
setRefClass("Dist")  
setRefClass("DistUniform", c("a", "b"), "Dist", methods = list(  
  mean <- function() {  
    (a + b) / 2  
  }  
))
```

---

#### # 1. way:

```
Person <- setRefClass("Person", methods = list(  
  say_hello = function() message("Hi!")  
))
```

#### # 2. way:

```
Person <- setRefClass("Person")  
Person$methods(say_hello = function() message("Hi!"))
```

#### Methods of object returned by setRefClass:

- **new** for creating new objects of that class. The new method takes named arguments specifying initial values for the fields
- **methods** for modifying existing or adding new methods
- **help** for getting help about methods
- **fields** to get a list of fields defined for class
- **lock** locks the named fields so that their value can only be set once