# Optional

Optional jest dostępna od wersji 1.8. **Optional** jest przygotowane z myślą o radzeniu sobie z sytuacją kiedy podczas runtime możemy dostać nulla. Optional opakowuje inny obiekt i mówi o nim, że może być nullem lub nie być nullem

## Metody obsługujące Optional

<https://github.com/idzikpro/JavaBasics/blob/master/src/main/java/pl/idzikpro/optional/OptionalMain.java>

Metody:

* metoda **Optional.offNullable(T)** -stosujemy wtedy gdy zakładamy, że obiekt przekazany jako T może być nullem
* metoda **Optional.of(T)**  - wtedy gdy wiemy, że obiekt przekazany jako T jest nullem
* metoda **Optional.empty()** – gdy chcemy zwrócić pustą wartość.

// case 0 - the value may be null - Optional.ofNullable()  
Student stMaybeNull = null;  
Optional<Student> studentMaybeNull = Optional.*ofNullable*(stMaybeNull);  
studentMaybeNull.ifPresent(System.*out*::println);  
  
//case 1 - the value can not be null - Optional.of()  
Student stNotNull = new Student("Jacek", 23);  
Optional<Student> studentNotNull = Optional.*of*(stNotNull);  
System.*out*.println(studentNotNull.get());  
  
  
//case 3 - the value is empty  
Optional<Student> stEmpty = Optional.*empty*();

## Korzystanie z Optional

Wykorzystujemy następujące przykładowe metody:

**ifPresent**(Consumer)- wykonywane gdy element istnieje

**ifPresentOrElse**(Consumer1,Consumer2) – jeśli element istnieje wykonuje się Consumer1, w przeciwnym przypadku Consumer2

**orElseThrow**(Exception Supplier) – jeśli nie ma obiektu, to rzuć wyjatkiem

**orElseGet**(Supplier) – jeśli istnieje, to go zwróci, a w przeciwnym przypadku wykona supplier (tutaj możemy uważać to za przekazanie wartości domyślnej)

// using Optional  
 List<Student> list = Arrays.*asList*(  
 new Student("Kasia", 22),  
 new Student("Jacek", 21),  
 new Student("Lucyna", 30)  
 );  
  
 Optional<Student> optionalAgeOver30Student = list.stream()  
 .filter(student -> student.getAge() > 30)  
 .findFirst();  
 //case - using ifPresent  
 optionalAgeOver30Student.ifPresent(st -> System.*out*.println(st));  
  
 //case - using ifPresentOrElse  
 optionalAgeOver30Student  
 .ifPresentOrElse(  
 st -> System.*out*.println(st),  
 () -> System.*out*.println("No such student")  
 );  
  
 //case - using orElseThrow  
 try {  
 optionalAgeOver30Student  
 .orElseThrow(NoSuchElementException::new);  
 } catch (NoSuchElementException e) {  
  
 }  
  
 //case - using orElseGet  
 System.*out*.println(optionalAgeOver30Student  
 .orElseGet(() -> new Student("Joanna", 31)));  
  
}

UWAGI: Optional nie jest serializowany.