

Software Ingeniaritza

KD: Klase diagrama

SD: Sekuentzia diagrama

Visual Paradigm

Software Ingeniaritza

Aurkibidea

Proiektu berria

KD: klaseak sortu / eraikitzalea / atributuak / metodoak

KD: Singleton

KD: Usage erlazioa

KD: Kodea sortu

SD: Aktorea sortu / instantziak / deiak

SD: Begiztak / baldintzak

Software Ingeniaritza

Aurkibidea

Proiektu berria

KD: klaseak sortu / eraikitzalea / atributuak / metodoak

KD: Singleton

KD: Usage erlazioak

KD: Kodea sortu

SD: Aktorea sortu / instantziak / deiak

SD: Begiztak / baldintzak

Visual Paradigm

Projektua sortu

1) Click New

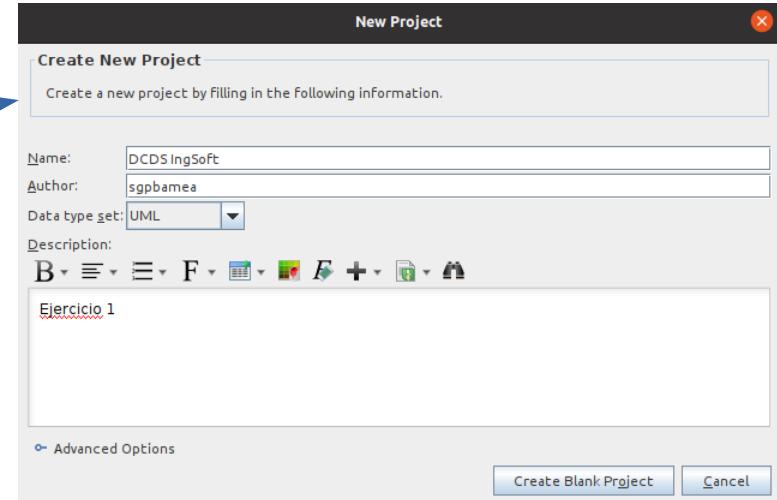


2) Click System Design → UML +



Class Diagram

Design object model, persistence model for Hibernate ORM and REST API with classes, their attributes, operations and inter-relationships.



Software Ingeniaritza

Aurkibidea

Proiektu berria

KD: klaseak sortu / eraikitzalea / atributuak / metodoak

KD: Singleton

KD: Usage erlazioak

KD: Kodea sortu

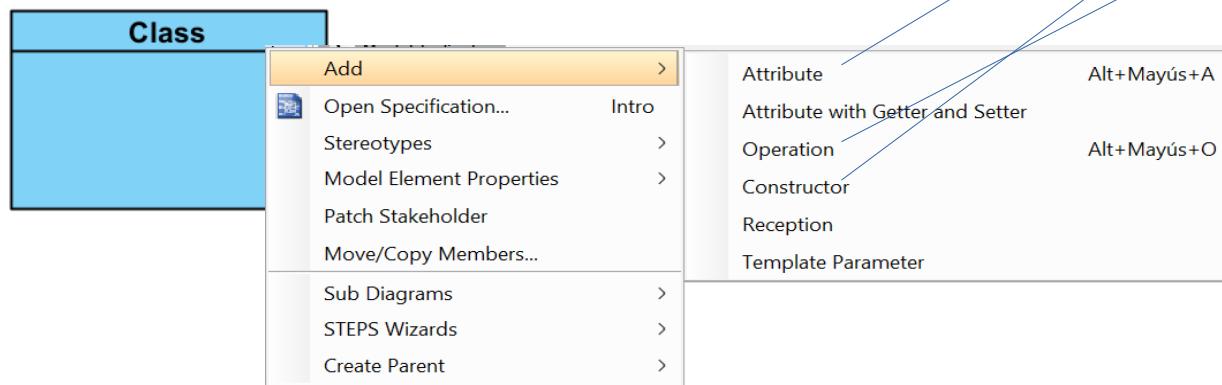
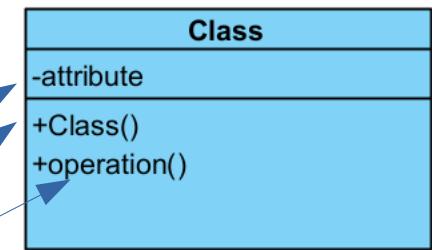
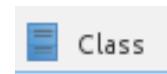
SD: Aktorea sortu / instantziak / deiak

SD: Begiztak / baldintzak

Kontuan izatekoak

Class Diagram barruan

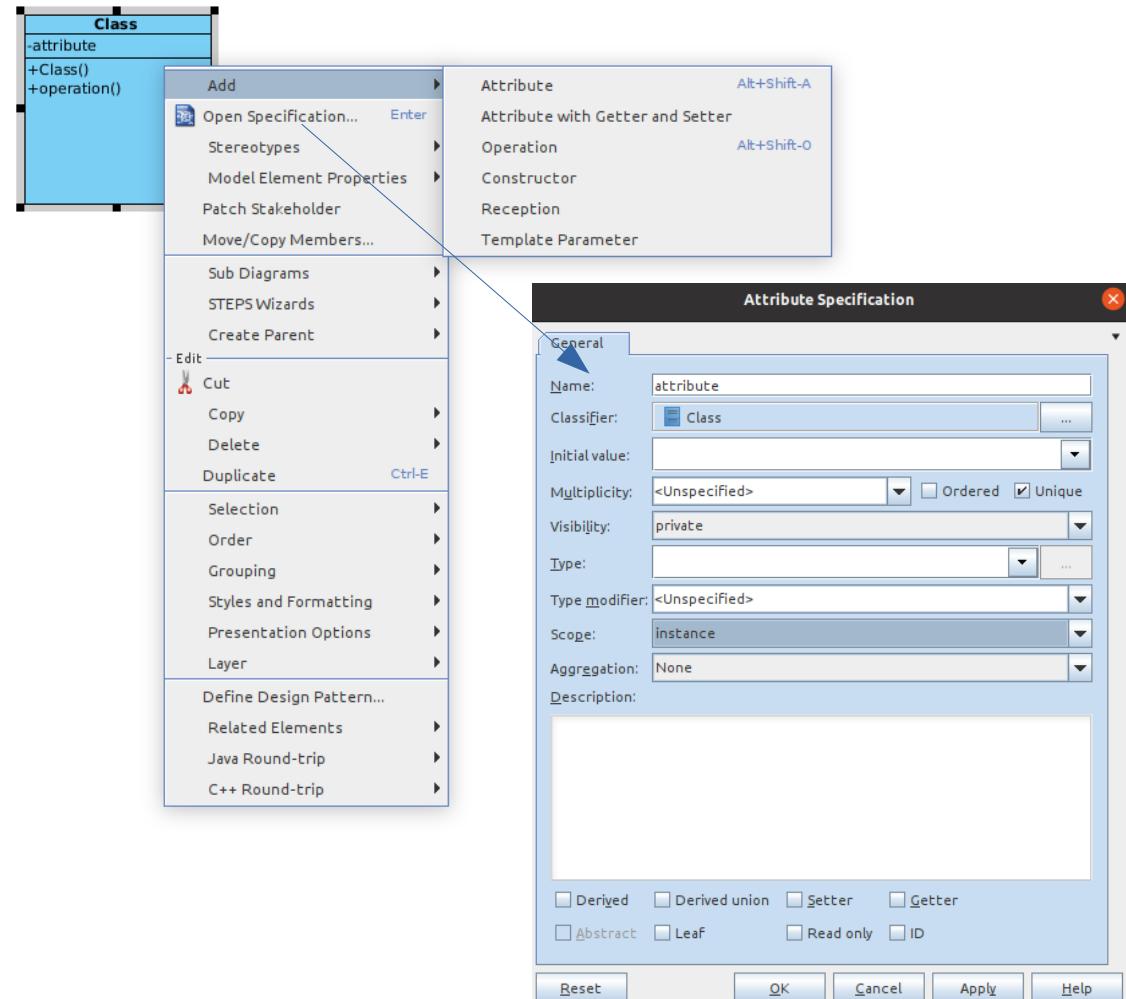
- Klasea sortu
- Sortu
 - Eraikitzzaileak (Add → Constructor)
 - Atributuak (Add → Attribute)
 - Metodoak (Add → Operation)



KD: Klaseak sortu/ eraikitzzailea / atributuak / metodoak

Class Diagram barruan

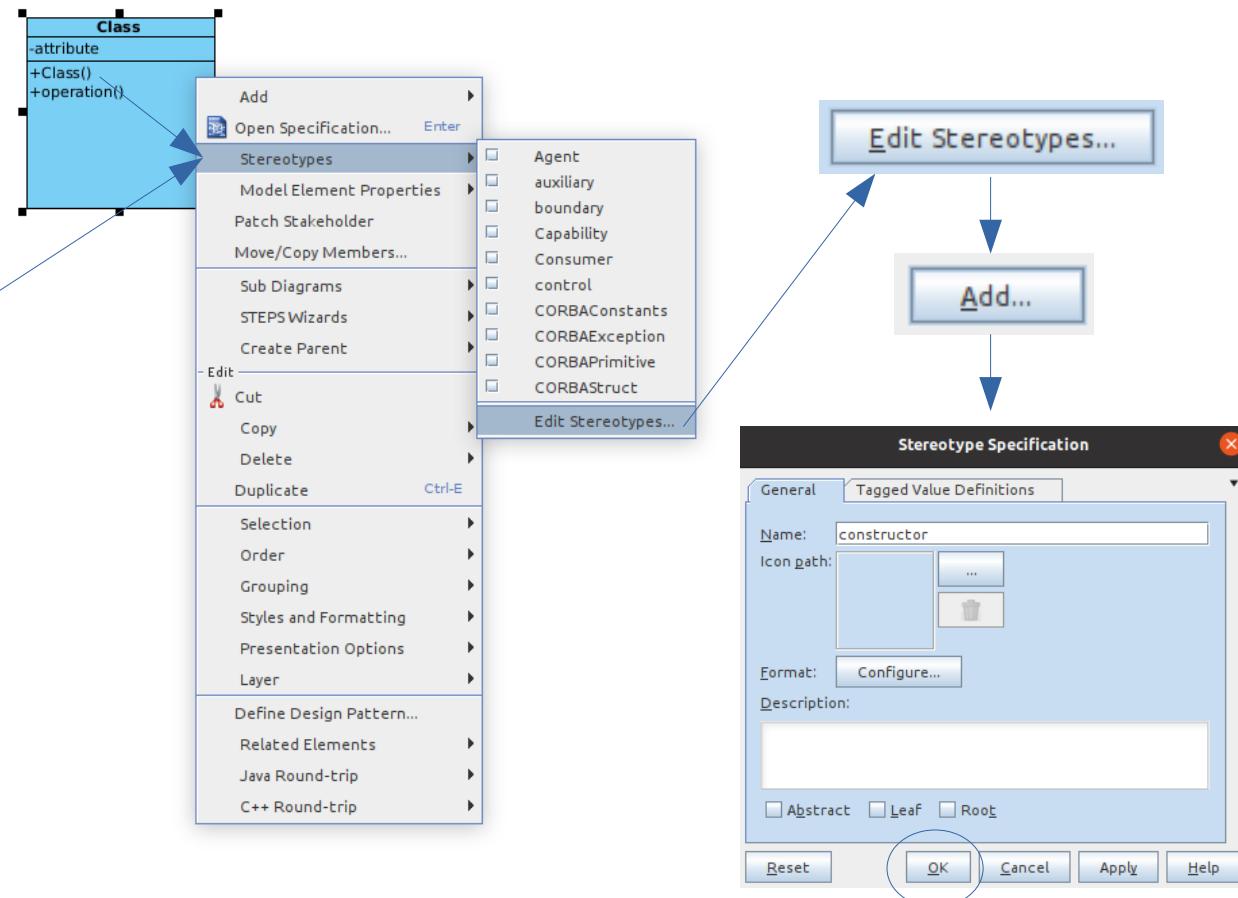
- Klasea sortu
- Sortu
 - **Atributuak**
 - **Eraikitzzailea**
 - **Metodoak (operation)**
- + public (Visibility)
- - private (Visibility)
- Static (Scope Classifier)



KD: Klaseak sortu/ eraikitzzailea / atributuak / metodoak

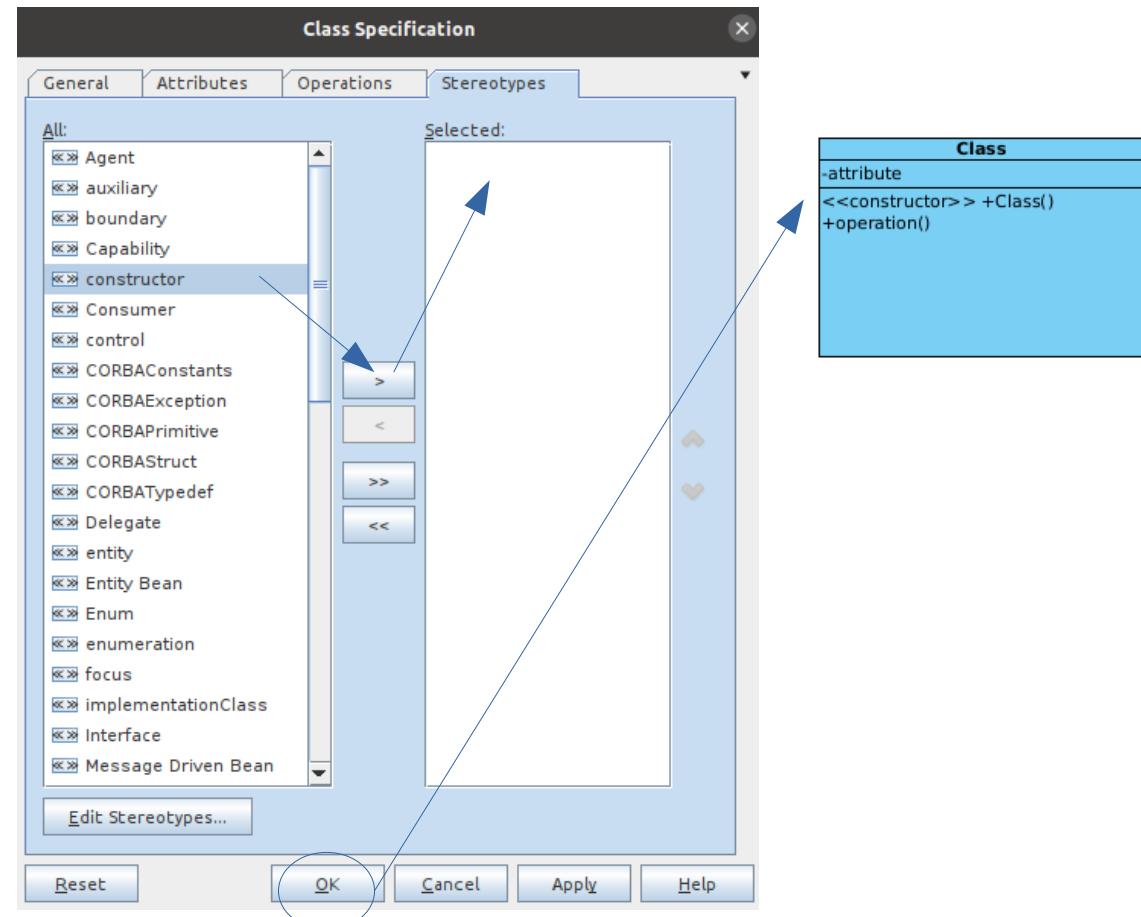
Class Diagram barruan

- Klasea sortu 
- Sortu
 - Atributuak
 - **Eraikitzzailea**
 - Metodoak (operation)
- + public (Visibility)
- - private (Visibility)
- Static (Scope Classifier)



Class Diagram barruan

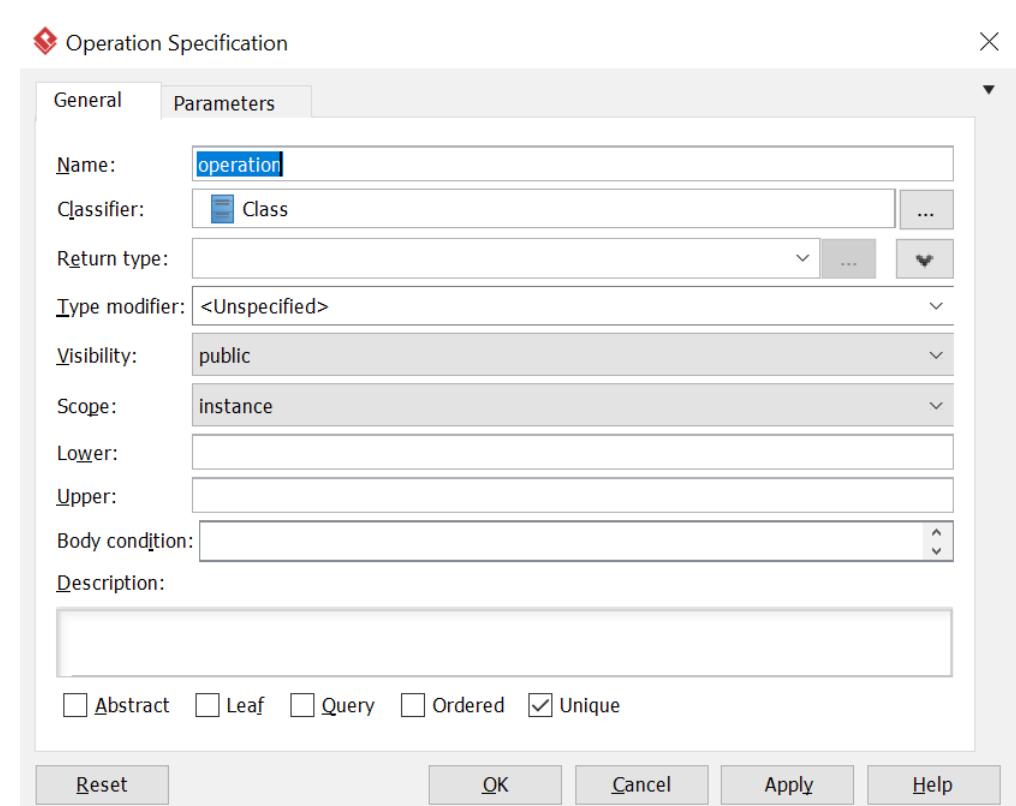
- Klasea sortu
- Sortu
 - Atributuak
 - **Eraikitzzailea**
 - Metodoak (operation)
- + public (Visibility)
- - private (Visibility)
- Static (Scope Classifier)



KD: Klaseak sortu/ eraikitzzailea / atributuak / metodoak

Class Diagram barruan

- Klasea sortu
- Sortu
 - Atributuak
 - Eraikitzzailea
 - **Metodoak**
- + public (Visibility)
- - private (Visibility)
- Static (Scope Classifier)



Software Ingeniaritza

Aurkibidea

Proiektu berria

KD: klaseak sortu / eraikitzalea / atributuak / metodoak

KD: Singleton

KD: Usage erlazioak

KD: Kodea sortu

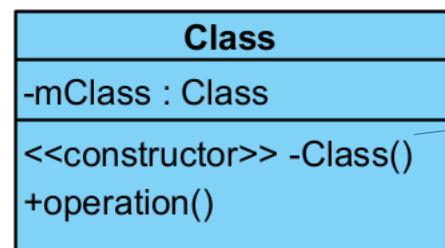
SD: Aktorea sortu / instantziak / deiak

SD: Begiztak / baldintzak

Kontuan izatekoak

Patroia

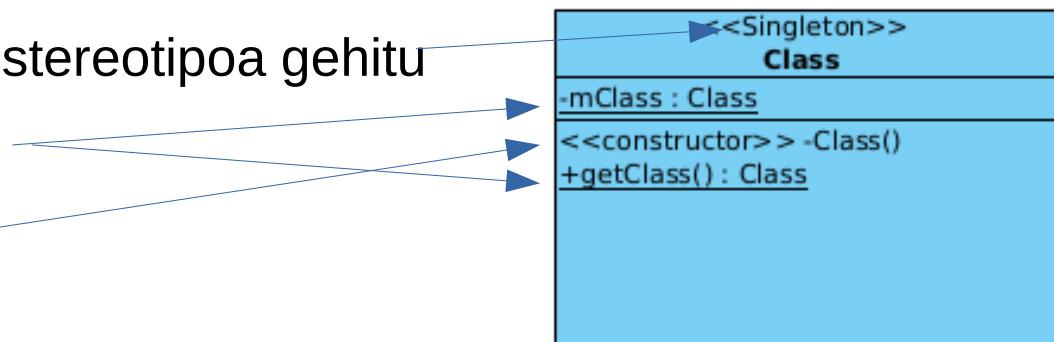
- **Helburua:** klase jakin batek instantzi bakarra izatea
→ Klasearen instantziaren sarbide globala



Adi, oraindik Singleton
ezaugarriak adierazi barik!

Patroia

- **Helburua:** klase jakin batek instantzi bakarra izatea
 - Klasearen instantziaren sarbide globala
- Pausuak:
 - Klaseari <<Singleton>> estereotipoa gehitu
 - Atributu eta getter estatikoak
 - Eraikitzale pribatua

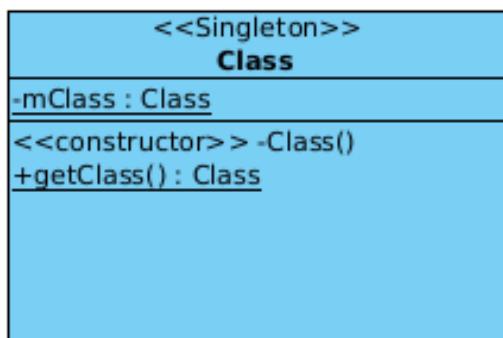


Orain bai!!!

KD: Singleton

Patroia

- **Helburua:** klase jakin batek instantzi bakarra izatea
 - Klasearen instantziaren sarbide globala



```
public class Class {
    private static Class mClass;
    private Class() {}
    public static Class getClass()
    {
        if (mClass==null)
        {
            mClass = new Class();
        }
        return mClass;
    }
}
```

Software Ingeniaritza

Aurkibidea

Proiektu berria

KD: klaseak sortu / eraikitzalea / atributuak / metodoak

KD: Singleton

KD: Usage erlazioak

KD: Kodea sortu

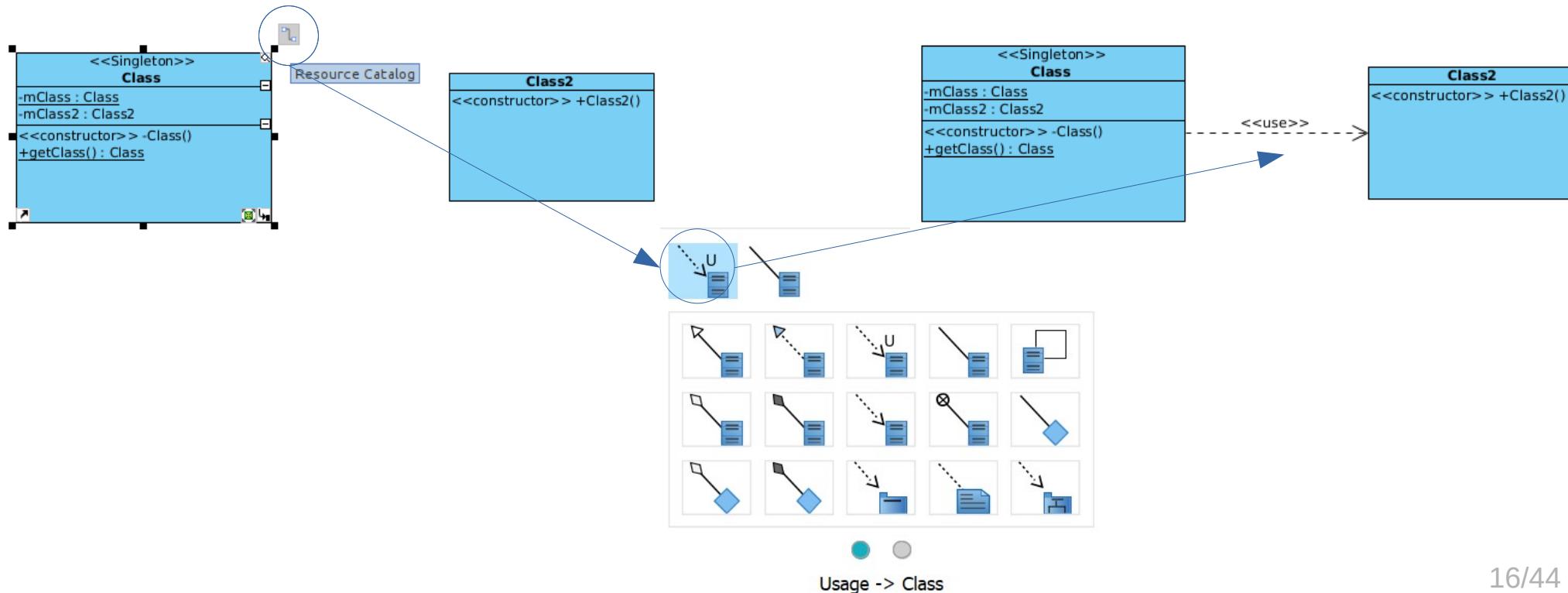
SD: Aktorea sortu / instantziak / deiak

SD: Begiztak / baldintzak

Kontuan izatekoak

KD: Usage erlazioak

Klaseen arteko *usage* erlazioak adierazteko



Software Ingeniaritza

Aurkibidea

Proiektu berria

KD: klaseak sortu / eraikitzalea / atributuak / metodoak

KD: Singleton

KD: Usage erlazioak

KD: Kodea sortu

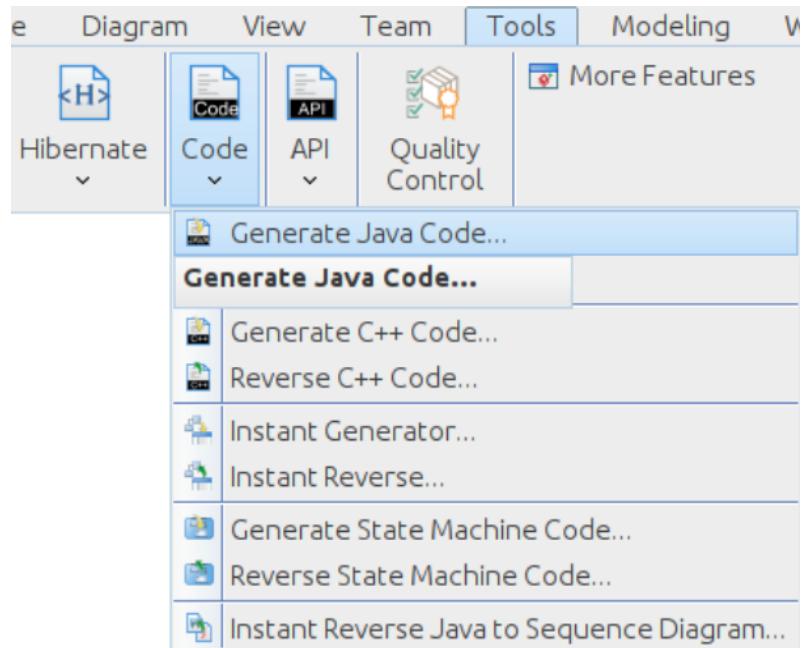
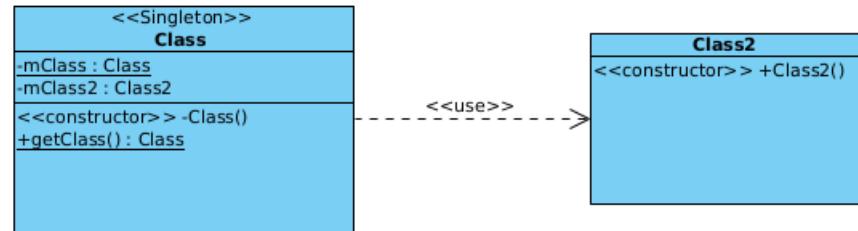
SD: Aktorea sortu / instantziak / deiak

SD: Begiztak / baldintzak

Kontuan izatekoak

KDtik habiatuta, kodea sortu

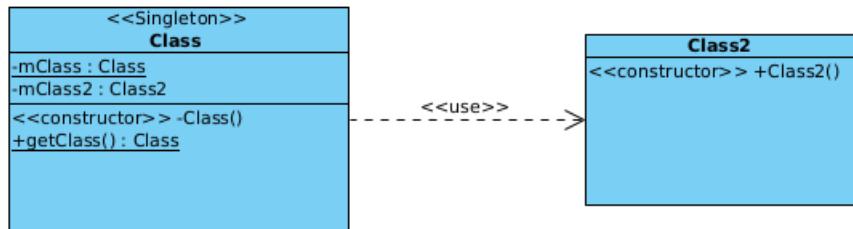
- Tools barruan
→ *Generate Java Code*
- Klase guztien .java fitxategiak sortzen ditu



KDtik habiatuta, kodea sortu

- Tools barruan
→ *Generate Java Code*
- Klase guztien .java fitxategiak sortzen ditu

```
public class Class {  
  
    private static Class mClass;  
    private Class2 mClass2;  
  
    private Class() {  
        // TODO - implement Class.Class  
        throw new UnsupportedOperationException();  
    }  
  
    public static Class getClass() {  
        // TODO - implement Class.getClass  
        throw new UnsupportedOperationException();  
    }  
}
```



```
public class Class2 {  
  
    public Class2() {  
        // TODO - implement Class2.Class2  
        throw new UnsupportedOperationException();  
    }  
}
```

Software Ingeniaritza

Aurkibidea

Proiektu berria

KD: klaseak sortu / eraikitzalea / atributuak / metodoak

KD: Singleton

KD: Usage erlazioak

KD: Kodea sortu

SD: Aktorea sortu / instantziak / deiak

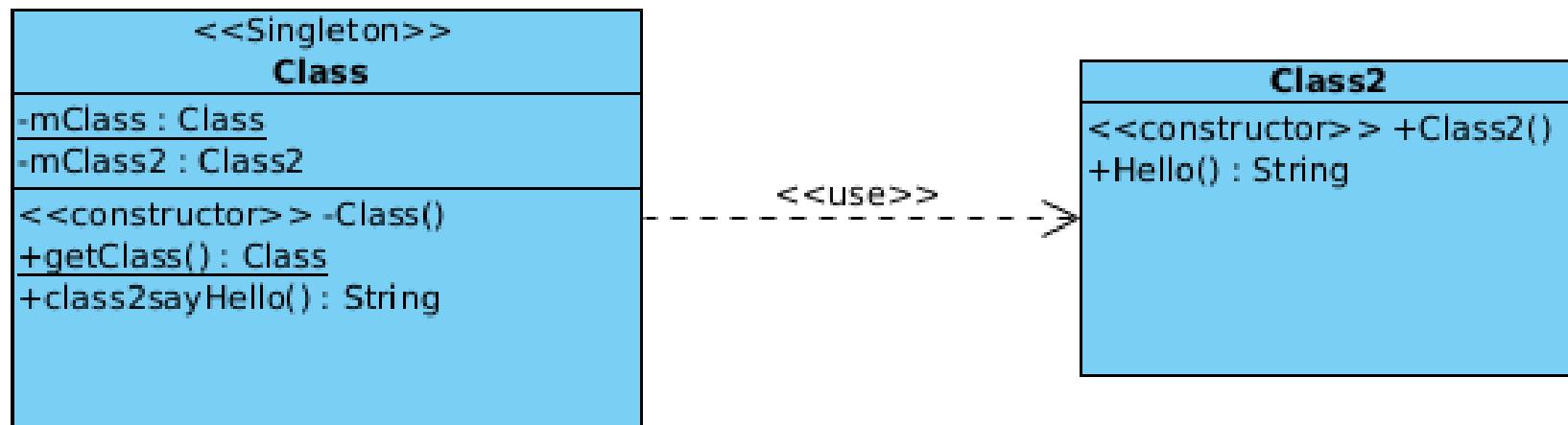
SD: Begiztak / baldintzak

Kontuan izatekoak

SD: Aktorea sortu / instantziak / deiak

Lehenik, gehitu funtziak klasiei:

- Class-ek Class2-ren instantzia du, eta azken horren Hello metodoari dei egiten dio

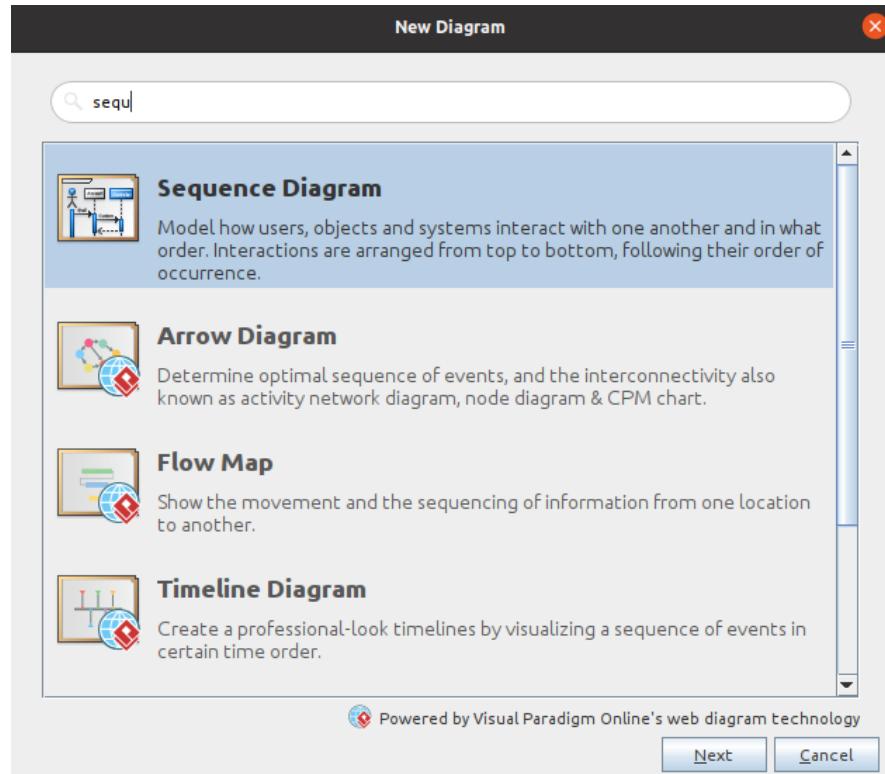


SD: Aktorea sortu / instantziak / deiak

Proiektuaren baitan, SD berria sortu

- *Diagrama barruan*
→ New

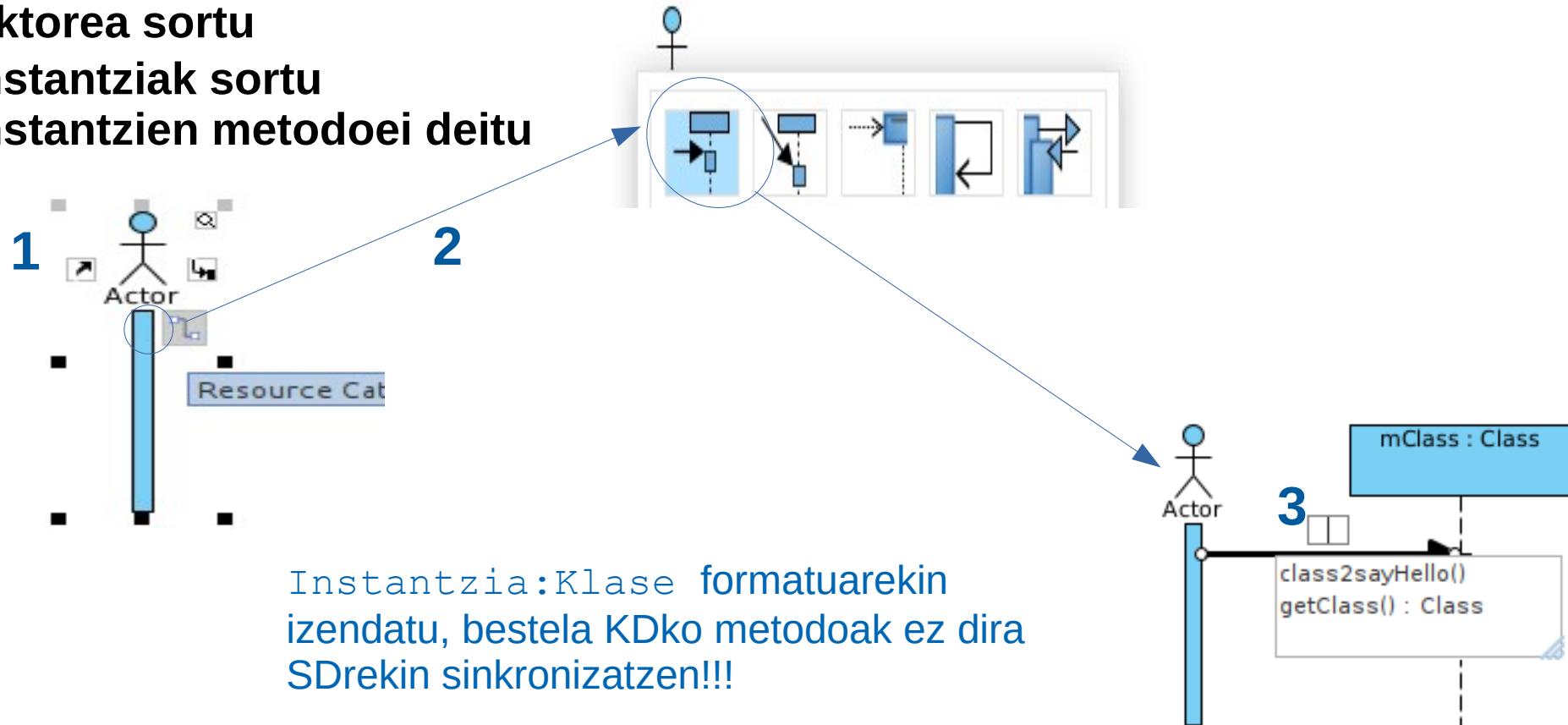
KDra bueltatzeko → *Switch diagrams*
(goian, eskubi aldean)



SD: Aktorea sortu / instantziak / deiak

Ekintzak:

- 1) Aktorea sortu**
- 2) Instantziak sortu**
- 3) Instantzien metodoei deitu**

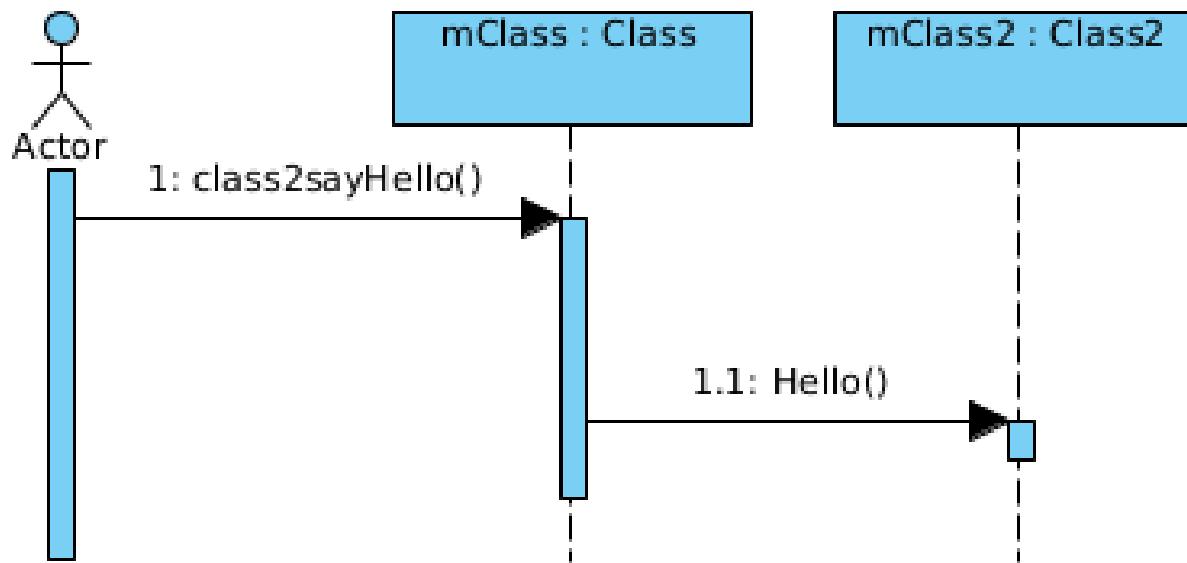


SD: Aktorea sortu / instantziak / deiak

Ekintzak:

- 1) Aktorea sortu**
- 2) Instantziak sortu**
- 3) Instantzien metodoei deitu**

•

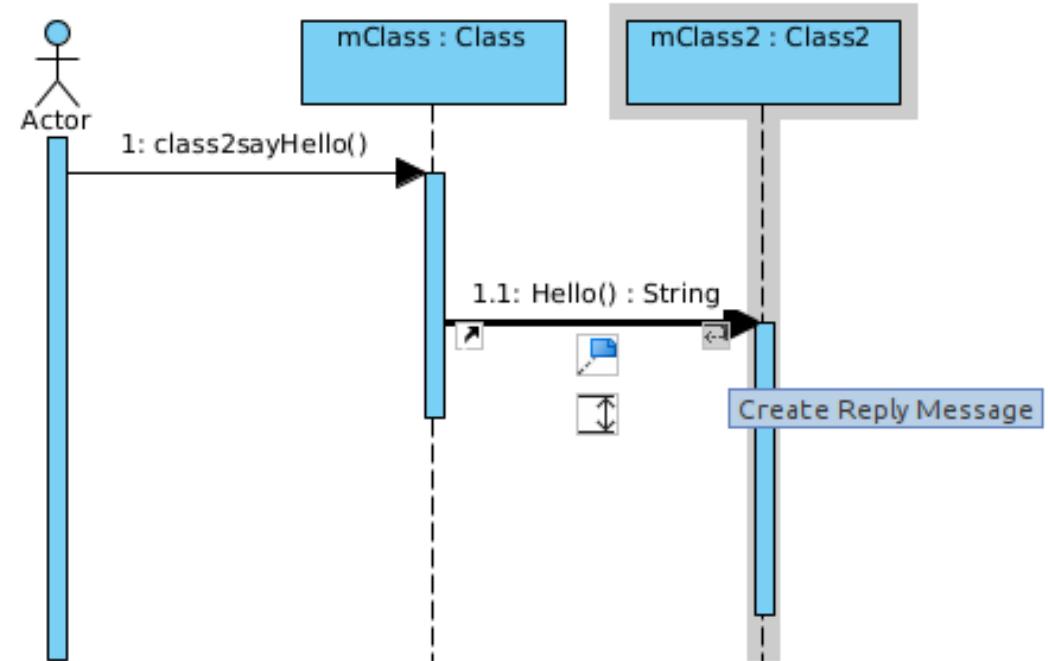


SD: Aktorea sortu / instantziak / deiak

Ekintzak:

- 1) Aktorea sortu
- 2) Instantziak sortu
- 3) Instantzien metodoei deitu

3.1 Metodoen bueltak adierazi

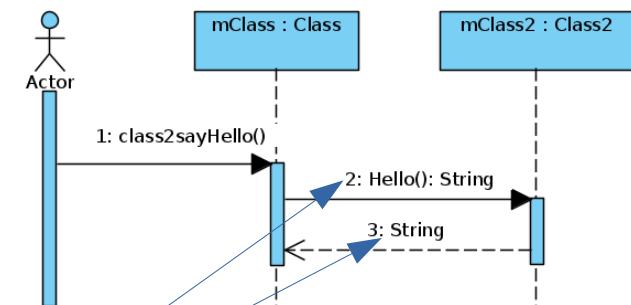
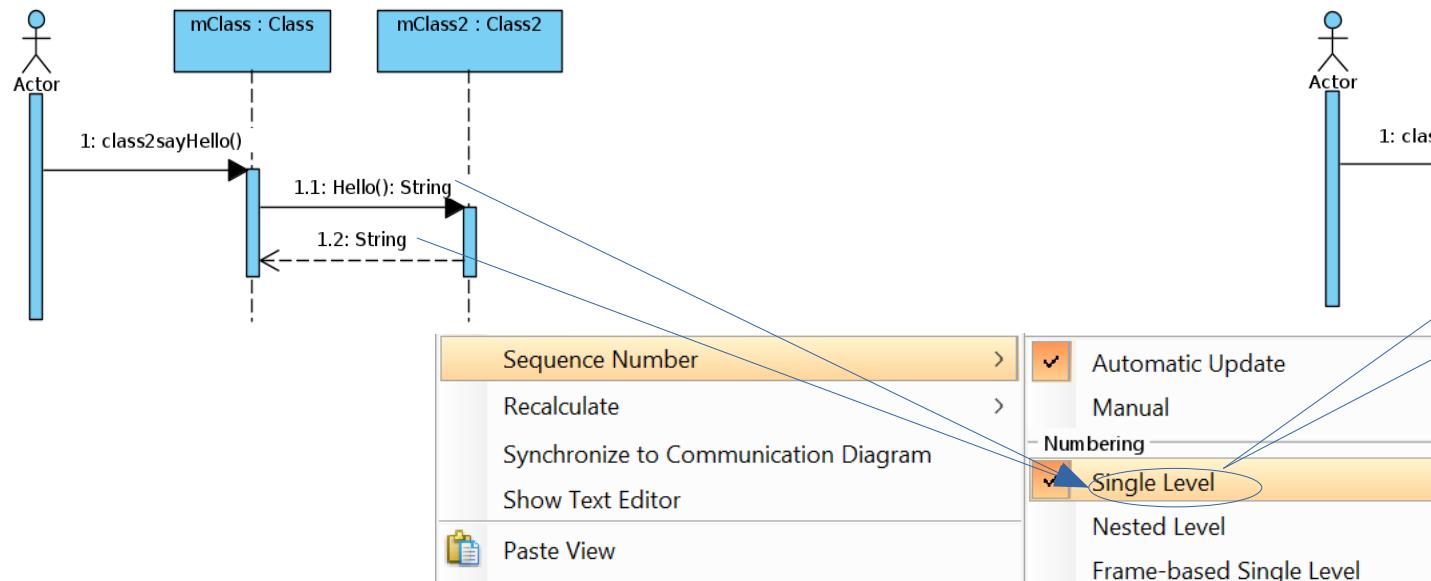


SD: Aktorea sortu / instantziak / deiak

Ekintzak:

- 1) Aktorea sortu
- 2) Instantziak sortu
- 3) Instantzien metodoei deitu
 - 3.1 Metodoen bueltak adierazi

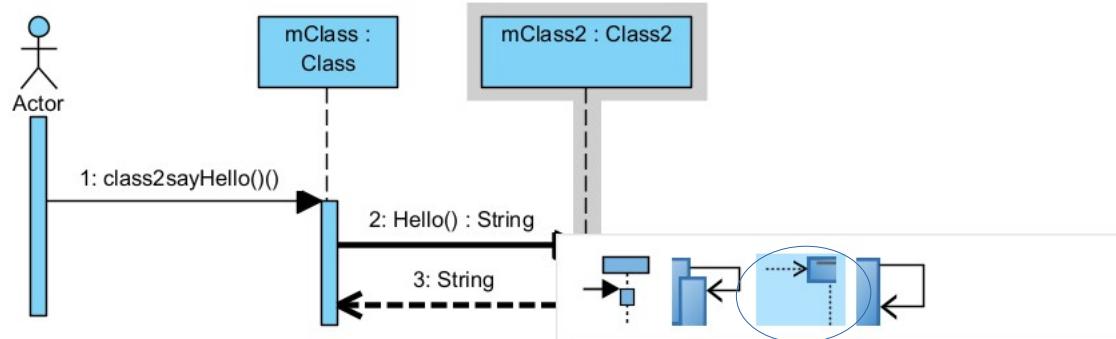
3.2 Metodoen deien numerazioa simplifikatu



SD: Aktorea sortu / instantziak / deiak

Ekintzak:

- 1) Aktorea sortu
- 2) Instantziak sortu
- 3) Instantzien metodoei deitu
 - 3.1 Metodoen bueltak adierazi
 - 3.2 Metodoen deien numerazioa simplifikatu
- 4) Klase baten instantzia berria sortu**

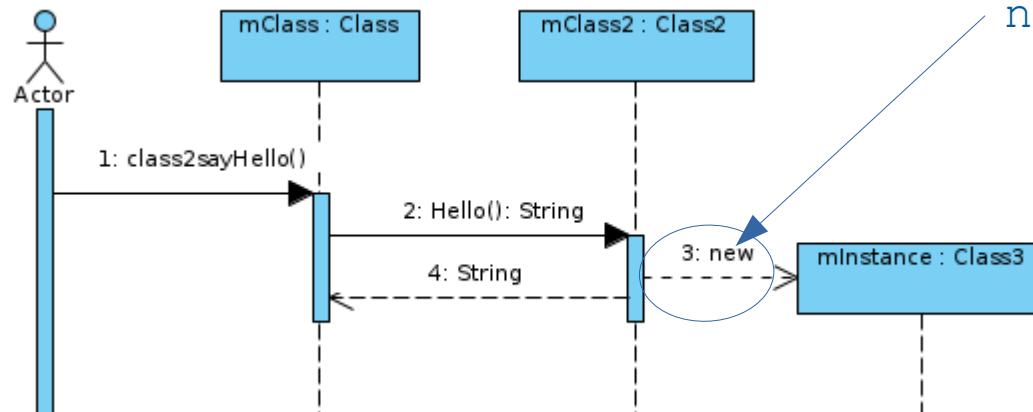


SD: Aktorea sortu / instantziak / deiak

Ekintzak:

- 1) Aktorea sortu
- 2) Instantziak sortu
- 3) Instantzien metodoei deitu
 - 3.1 Metodoen bueltak adierazi
 - 3.2 Metodoen deien numerazioa simplifikatu
- 4) Klase baten instantzia berria sortu

Instantzia berriaren deian
new idatzi!!



Software Ingeniaritza

Aurkibidea

Proiektu berria

KD: klaseak sortu / eraikitzalea / atributuak / metodoak

KD: Singleton

KD: Usage erlazioak

KD: Kodea sortu

SD: Aktorea sortu / instantziak / deiak

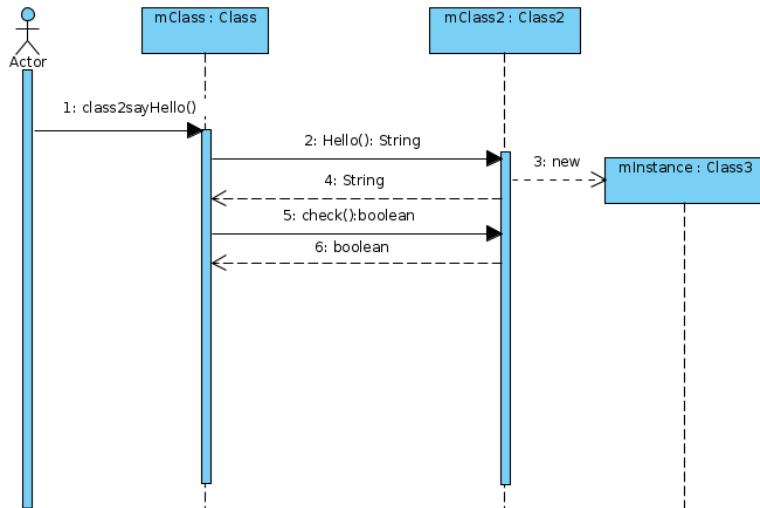
SD: Begiztak / baldintzak

Kontuan izatekoak

SD: Begiztak / baldintzak

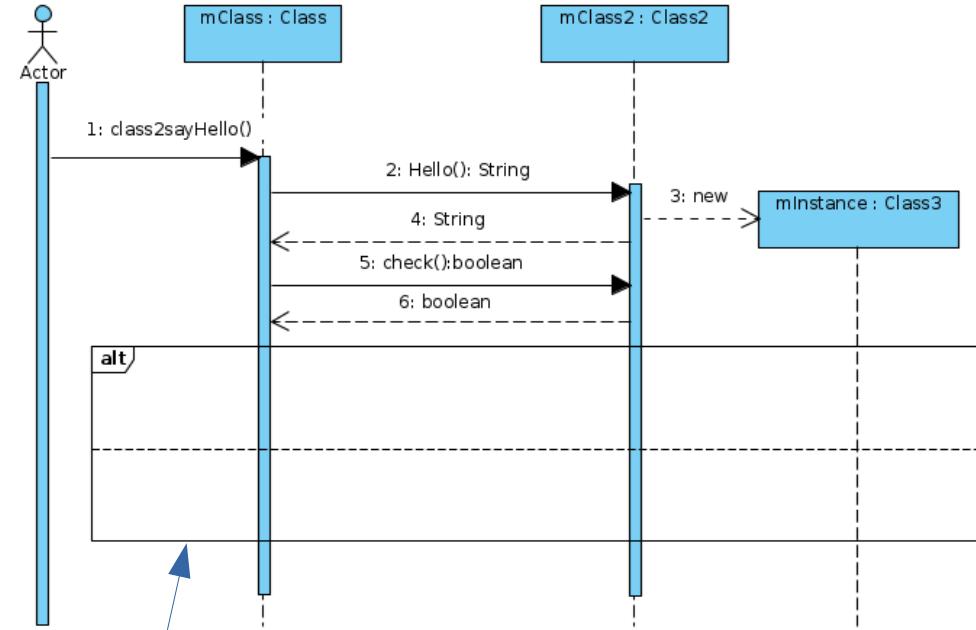
Ekintzak:

1) Baldintzak sortu



Alt. Combined Fragment

Alt. Combined Fragment
baliabidearen bitartez, if-else
moduko baldintza sortu

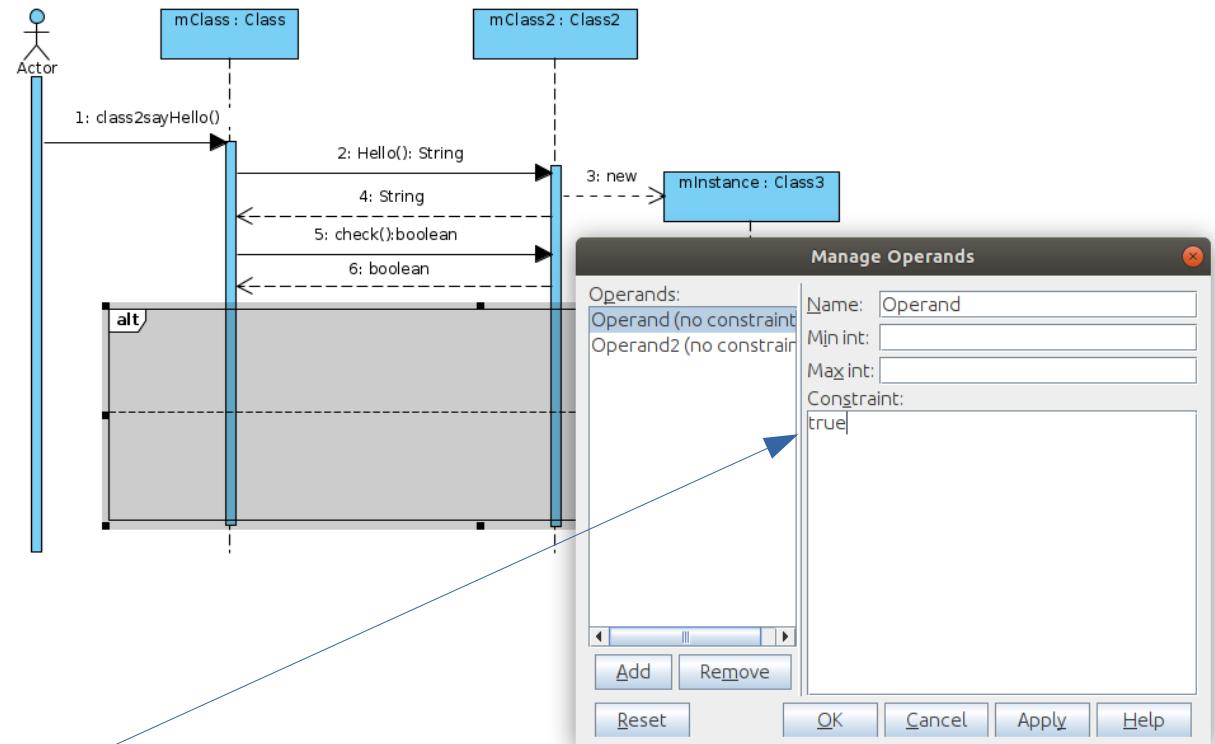


SD: Begiztak / baldintzak

Ekintzak:

1) Baldintzak sortu

1.1 Baldintza definitu



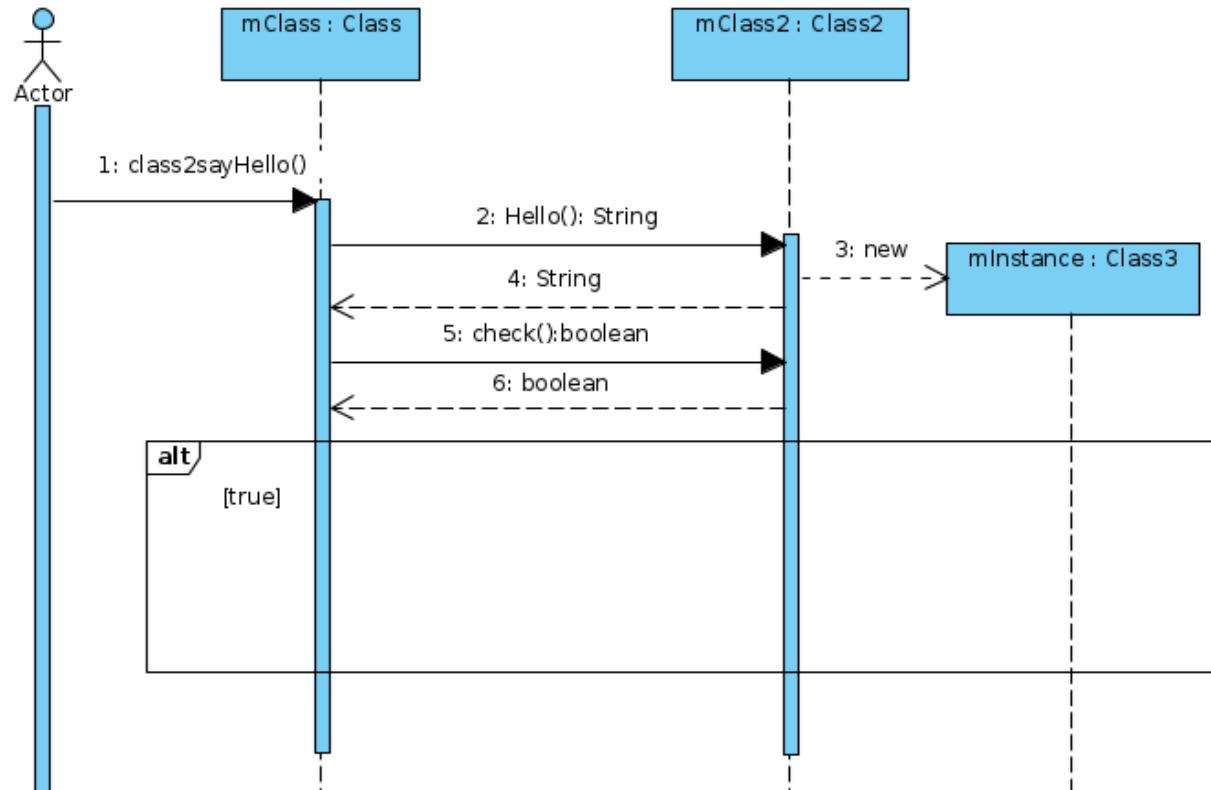
Baldintzan eskuin click-a egin,
Operand → Edit Operand aukeratu,
eta baldintza definitu.

Esaterako, check-en buelta true izatea

SD: Begiztak / baldintzak

Ekintzak:

- 1) Baldintzak sortu
 - 1.1 Baldintza definitu
 - 1.2 *Operand* ezabatu**



Baldintzan eskuin click-a egin,
Operand → *Remove Operand*, eta
Ezabati nahi den Operand aukera

Kasu hontan, `else-ri` dagokiona

SD: Begiztak / baldintzak

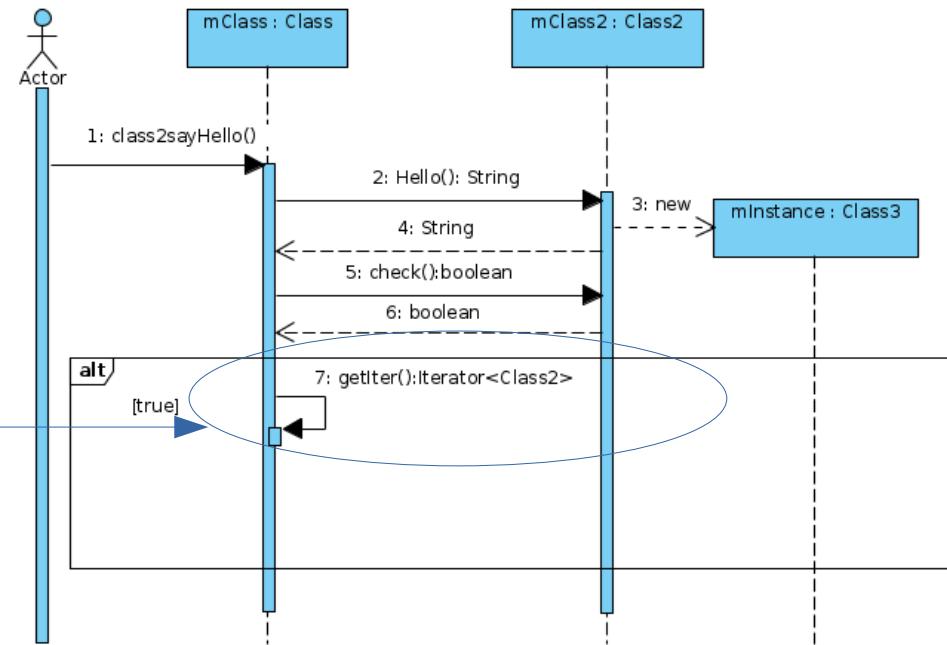
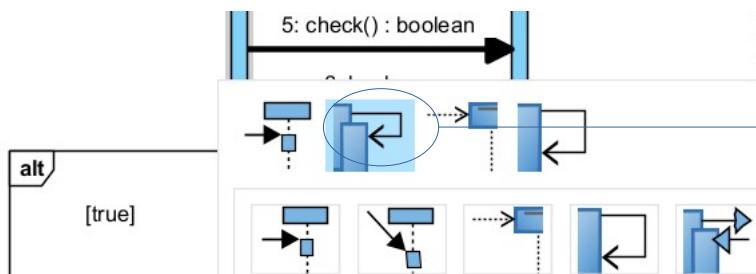
Ekintzak:

1) Baldintzak sortu

- 1.1 Baldintza definitu
- 1.2 *Operand* ezabatu

2) Begiztak sortu

2.1 Iterator pribatua

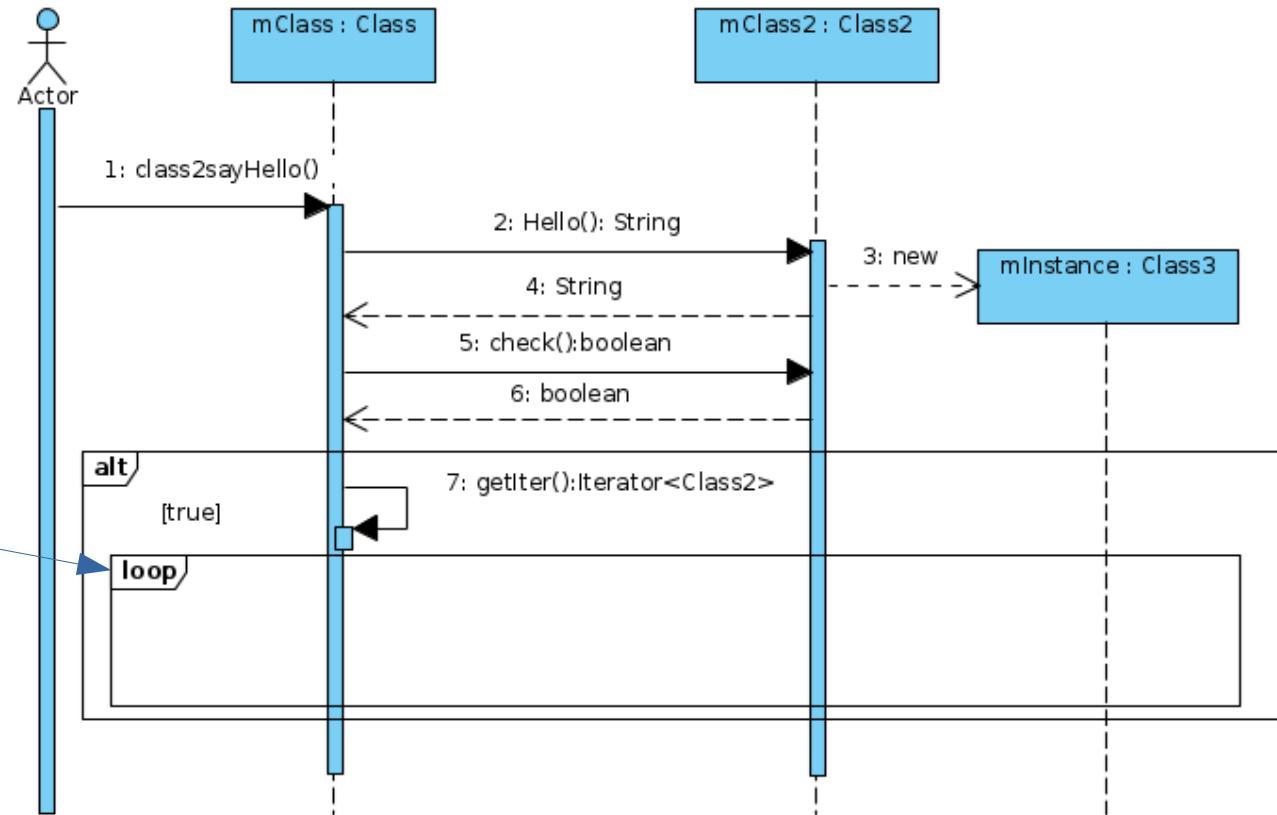


SD: Begiztak / baldintzak

Ekintzak:

- 1) Baldintzak sortu
 - 1.1 Baldintza definitu
 - 1.2 *Operand* ezabatu
- 2) Begiztak sortu
 - 2.1 Iterator pribatua
 - 2.2 Begizta adierazi**

LoopCombinedFragment
baliabidea erabili



SD: Begiztak / baldintzak

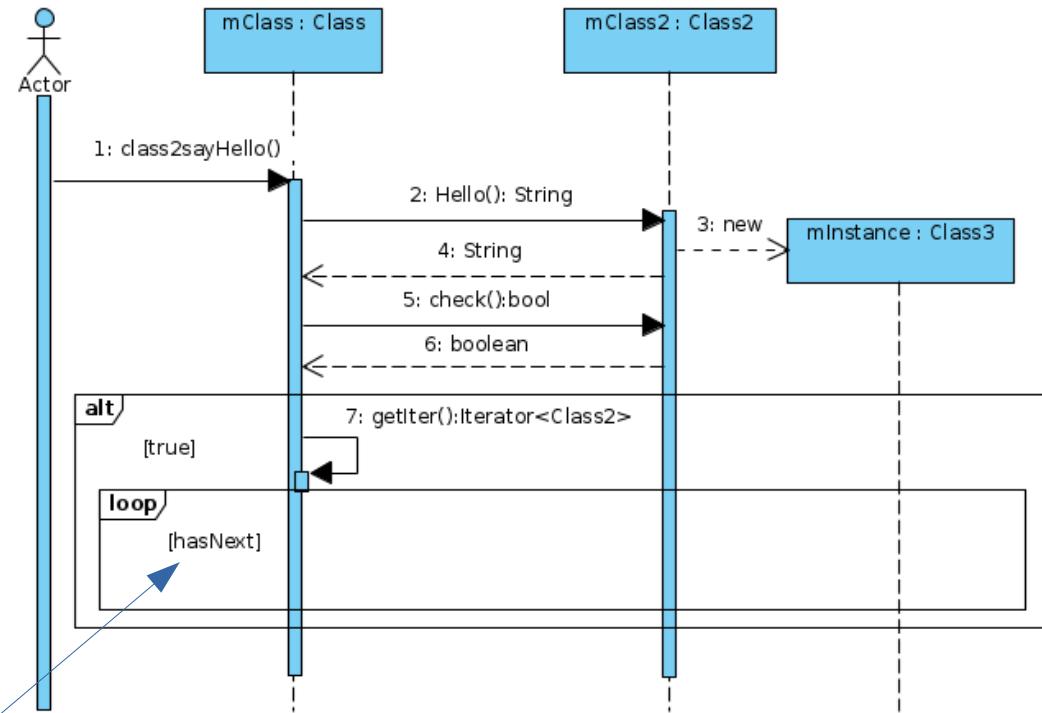
Ekintzak:

- 1) Baldintzak sortu
 - 1.1 Baldintza definitu
 - 1.2 *Operand* ezabatu

- 2) Begiztak sortu
 - 2.1 Iterator pribatua
 - 2.2 Begizta adierazi
 - 2.3 Iterazio baldintza definitu**

Begiztan eskuin click-a egin,
Operand → *Edit Operand* aukeratu,
eta iterazio baldintza definitu.

Orohar, iteratu `hasNext` dagoen bitartean



Software Ingeniaritza

Aurkibidea

Proiektu berria

KD: klaseak sortu / eraikitzalea / atributuak / metodoak

KD: Singleton

KD: Usage erlazioak

KD: Kodea sortu

SD: Aktorea sortu / instantziak / deiak

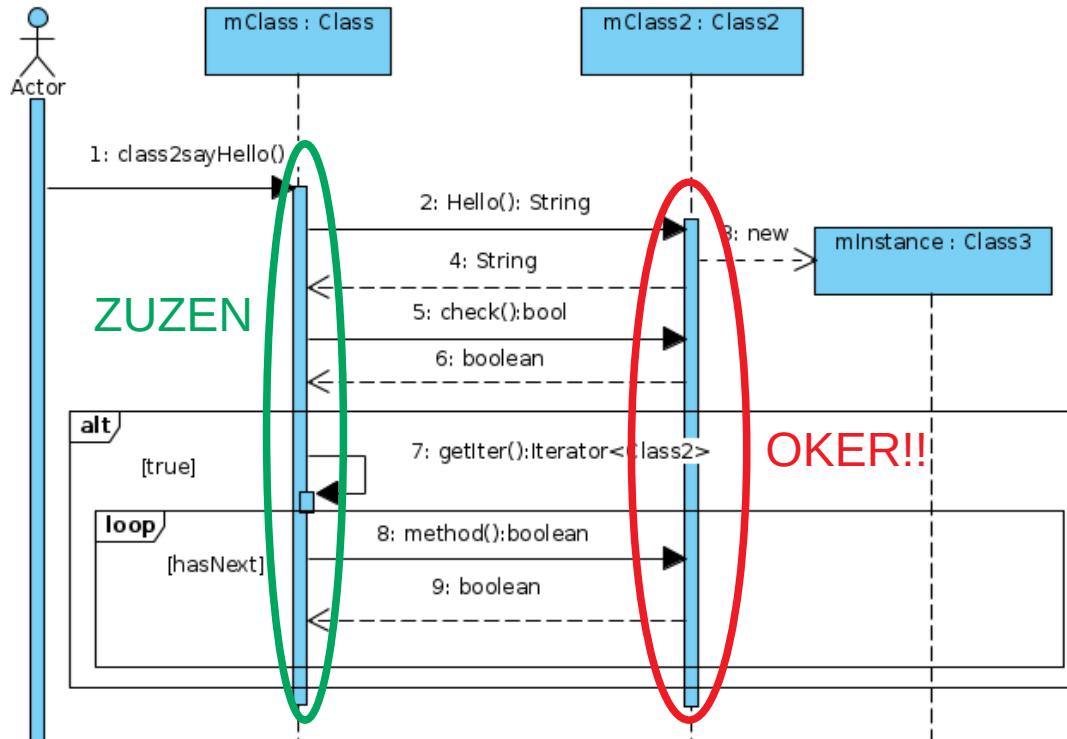
SD: Begiztak / baldintzak

Kontuan izatekoak

Kontuan izatekoak

Kontuan izateko:

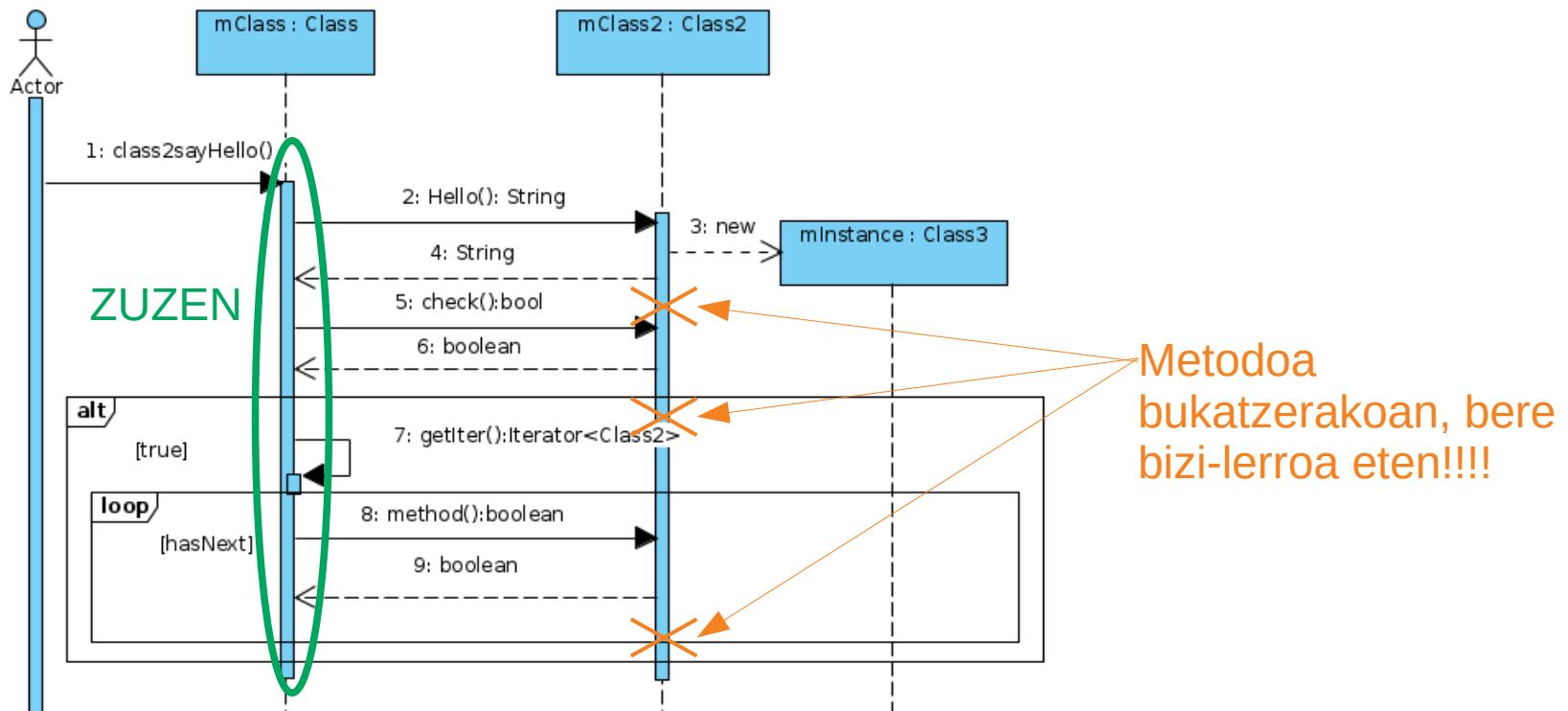
1) Metodoen bizi-erroek metodoen iraupena!



Kontuan izatekoak

Kontuan izateko:

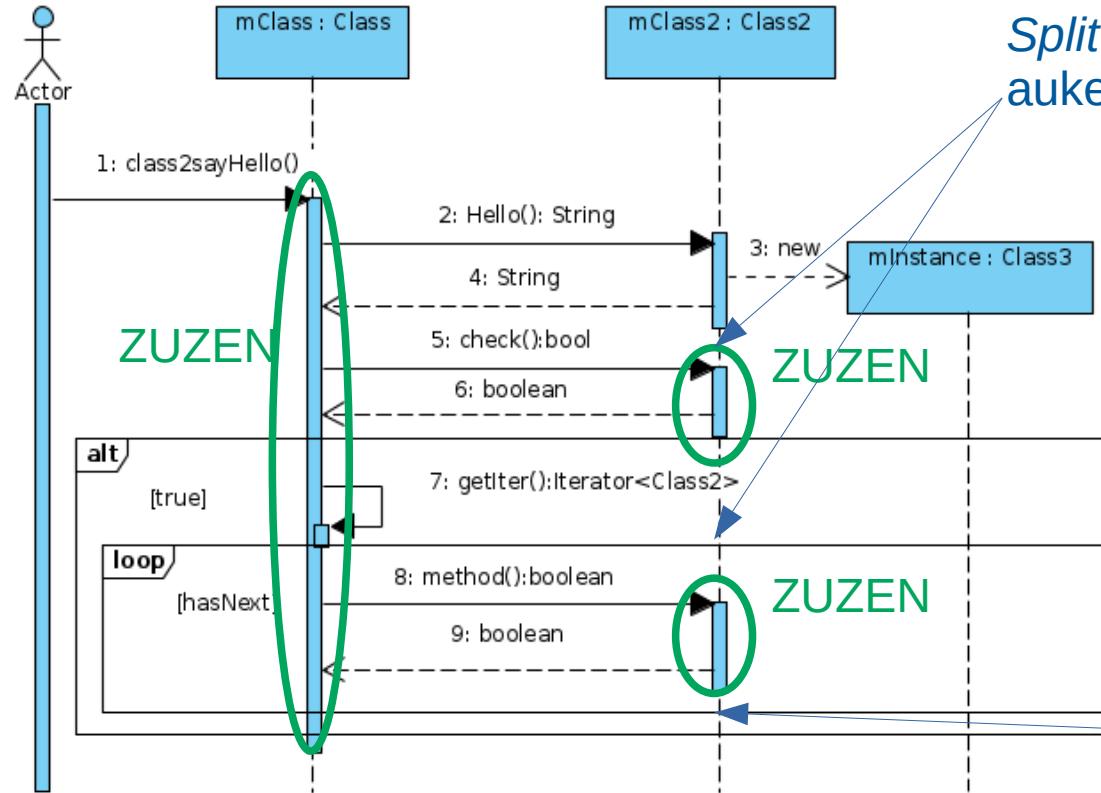
1) Metodoen bizi-lerroek metodoen iraupena!



Kontuan izatekoak

Kontuan izateko:

1) Metodoen bizi-lerroek metodoen iraupena!



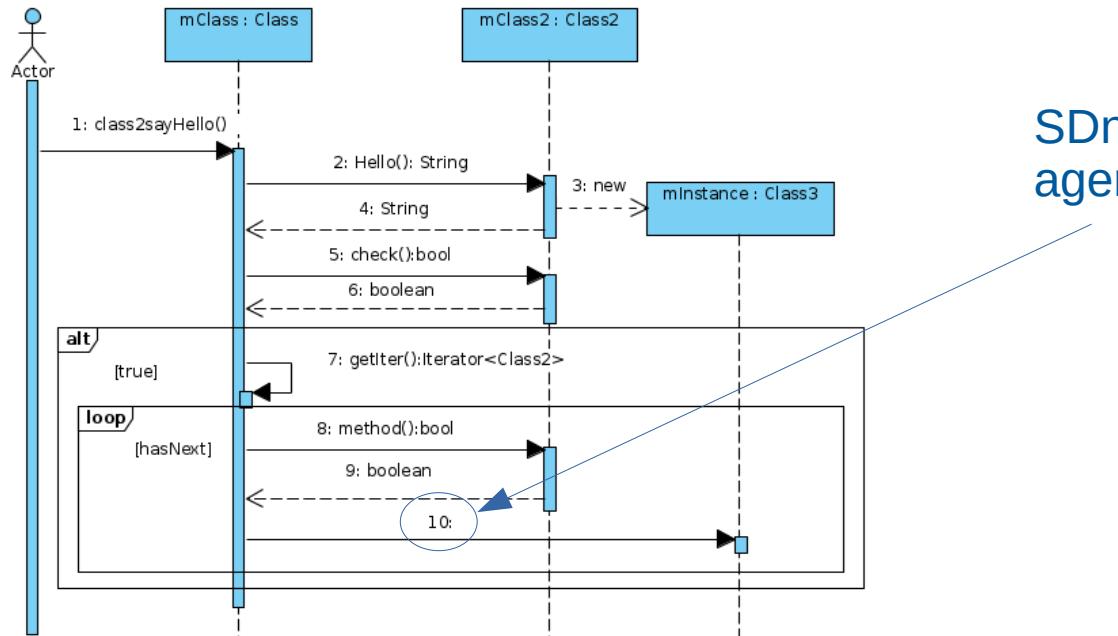
Bizi-lerroan eskuin click-a eta
Split Overlapping Execution
aukeratu. Bizi-lerroa zatitu.

Bizi-lerroen luzera
egokitu metodoen
iraupenera

Kontuan izatekoak

Kontuan izateko:

- 1) Metodoen bizi-erroek metodoen iraupena!
- 2) **KD eta SD metodoen sinkronizazioa**

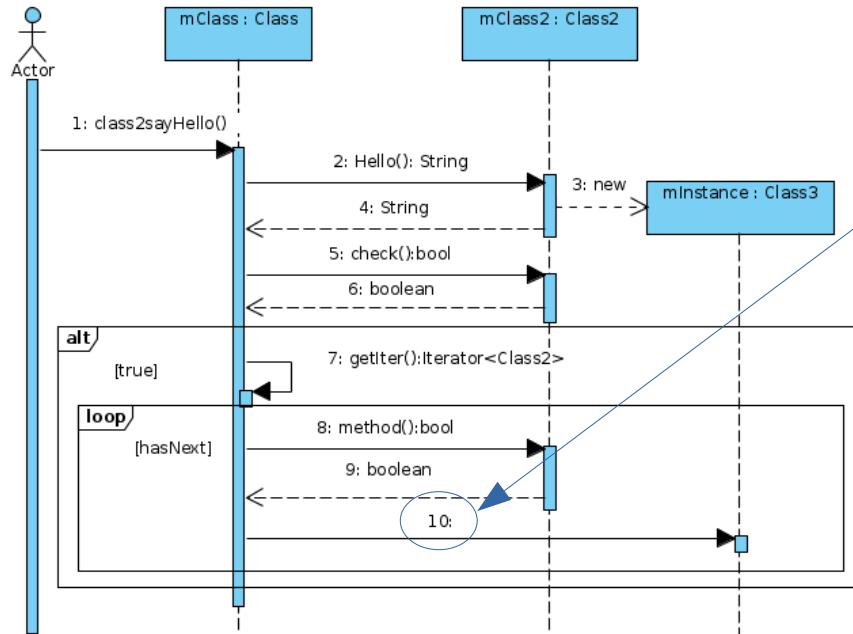


SDn metodo bat sortu eta KDn ere agertzeko, nola egin!?

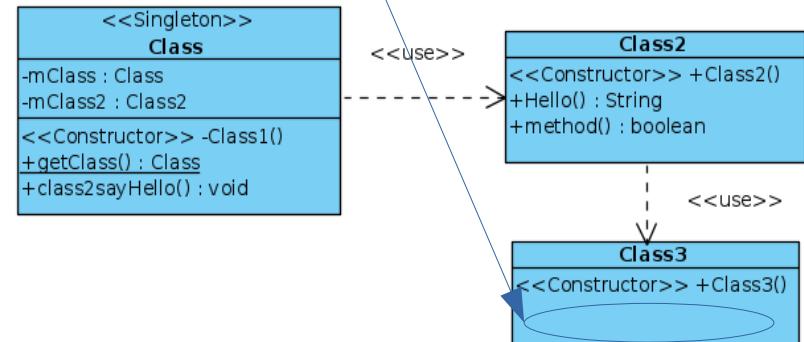
Kontuan izatekoak

Kontuan izateko:

- 1) Metodoen bizi-lerroek metodoen iraupena!
- 2) **KD eta SD metodoen sinkronizazioa**



SDn metodo bat sortu eta KDn ere agertzeko, nola egin!?!?!



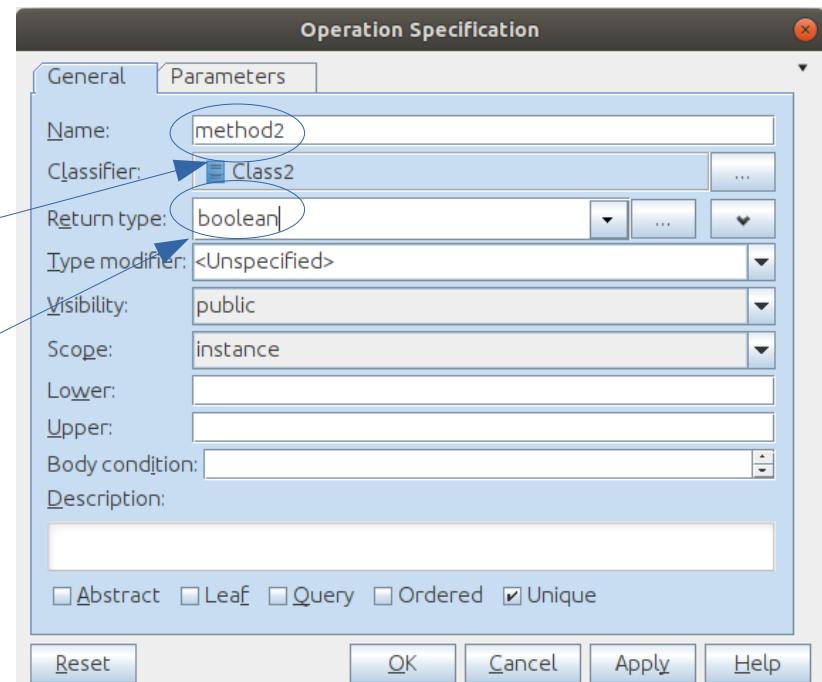
Kontuan izatekoak

Kontuan izateko:

- 1) Metodoen bizi-lerroek metodoen iraupena!
- 2) **KD eta SD metodoen sinkronizazioa**

Metodoaren deian eskuin click-a eta
Type(unspecified) → Call → Create Operation...

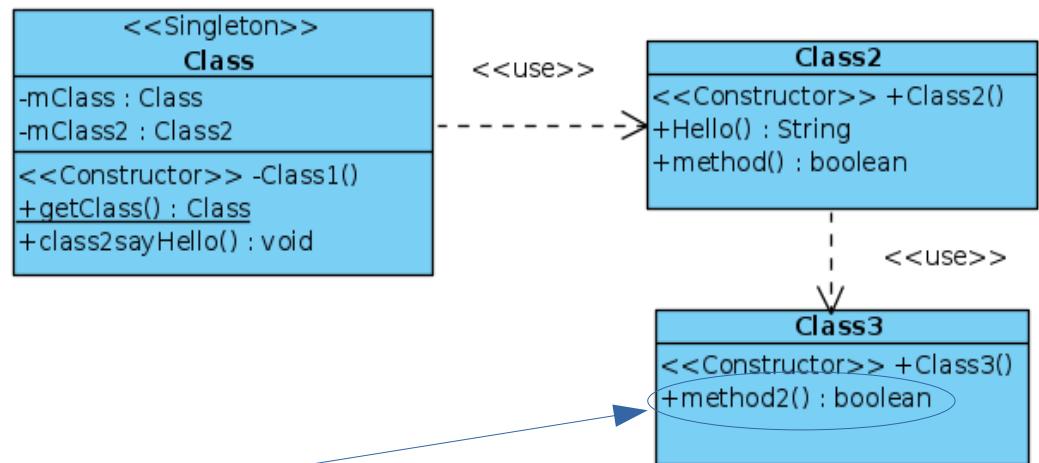
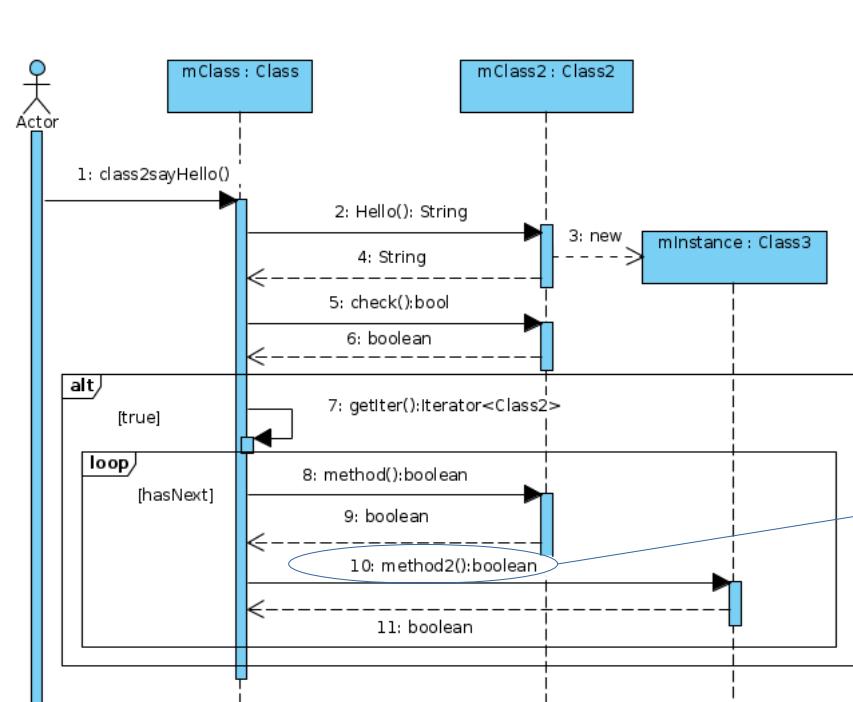
Esaterako, method2 izenekoa eta boolean
buelta daukana. *Parameters* atalean
parametroak sartzeko aukera.



Kontuan izatekoak

Kontuan izateko:

- 1) Metodoen bizi-lerroek metodoen iraupena!
- 2) KD eta SD metodoen sinkronizazioa



*SDn metodo berri bat sortuta,
KDn automakikoki eguneratu.*

Ingeniería del Software

Galderak

