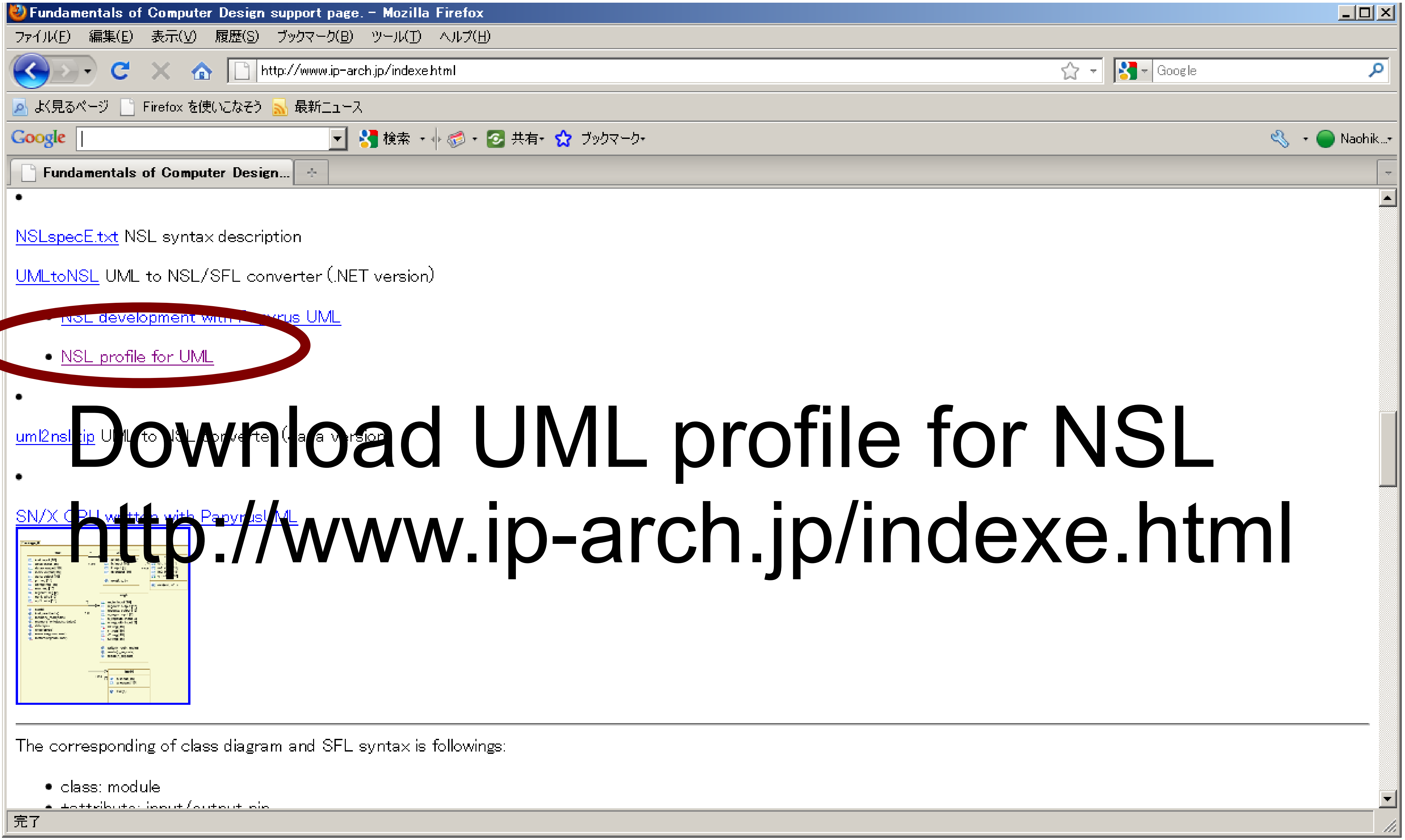
# NSL development with Papyrus UML

30/03/2010

IP ARCH, Inc.
Naohiko Shimizu, Ph.D.



Fundamentals of Computer Design support page. - Mozilla Firefox

http://www.ip-arch.jp/indexe.html

•

NSLspecE.txt NSL syntax description

UMLtoNSL UML to NSL/SFL converter (.NET version)

- NSL development with Papyrus UML
- NSL profile for UML

•

uml2nsl.zip UML to NSL converter (Java version)

•

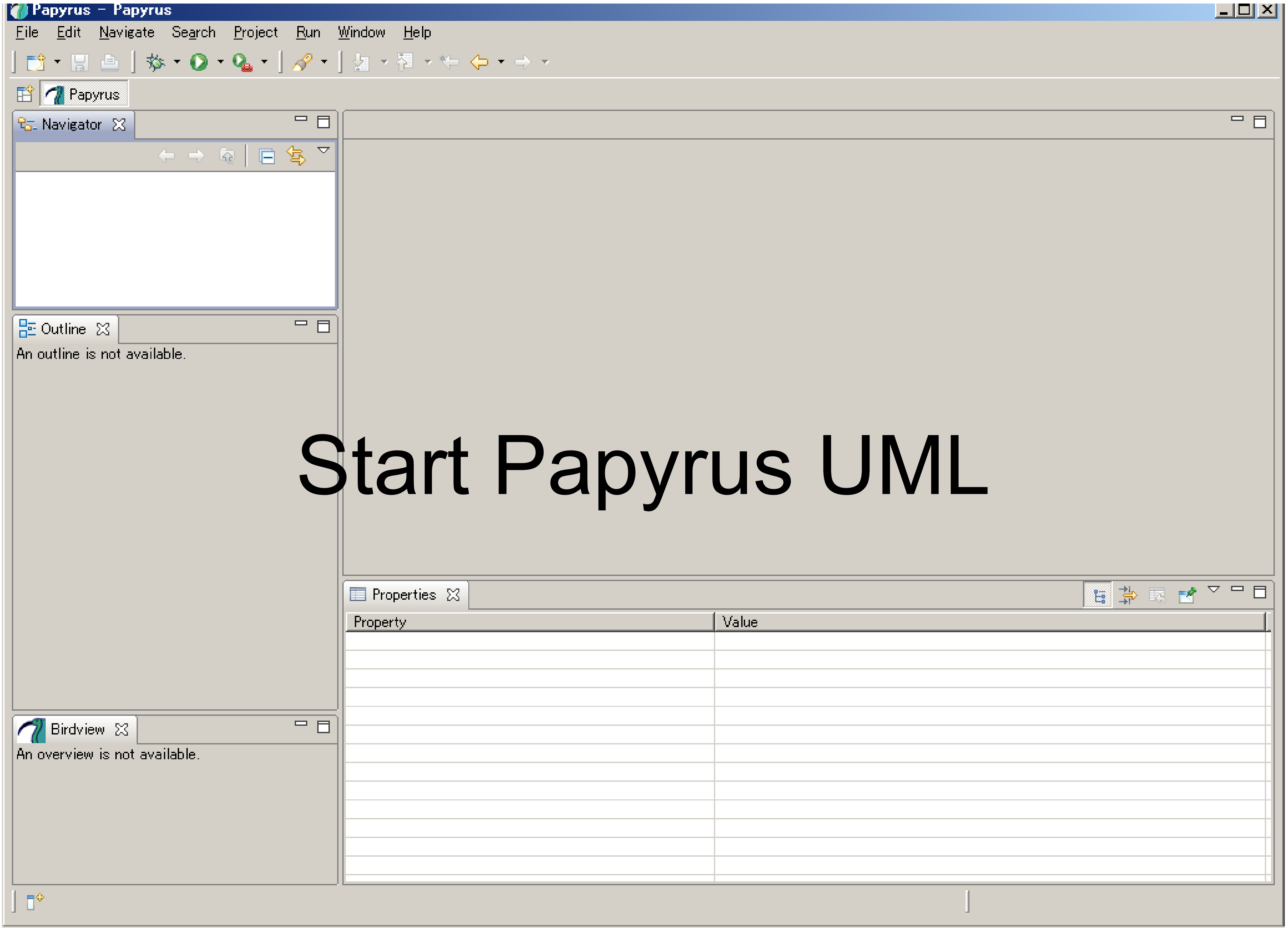
SN/X CPU written with PapyrusUML

The corresponding of class diagram and SFL syntax is followings:

- class: module
- +attribute: input/output pin

ファイル(F) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T) ヘルプ(H)
よく見るページ
Firefox を使いこなそう
最新ニュース
検索
共有
ブックマーク
Naohik...
Fundamentals of Computer Design...
完了

# Download UML profile for NSL
# http://www.ip-arch.jp/indexe.html

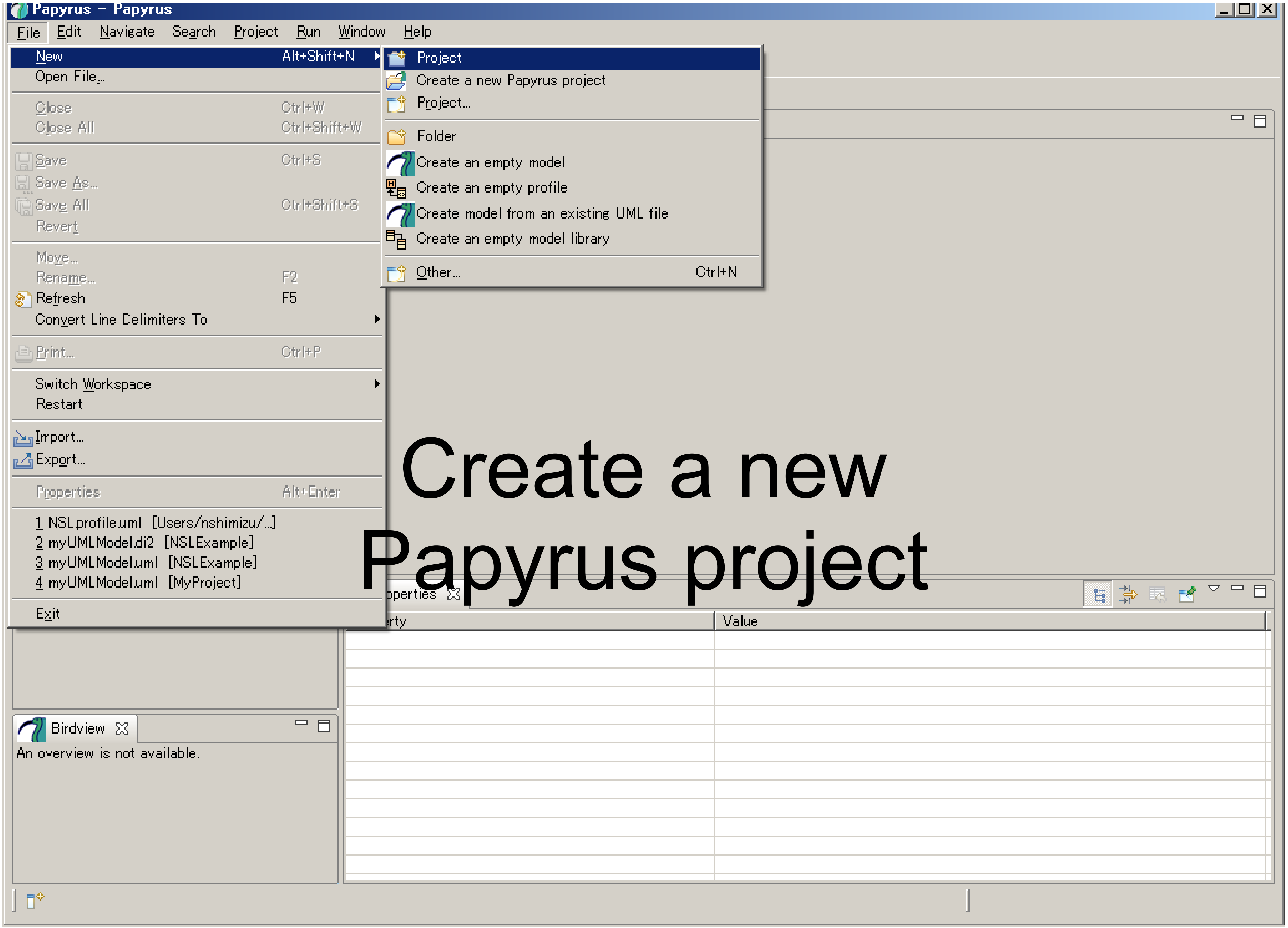


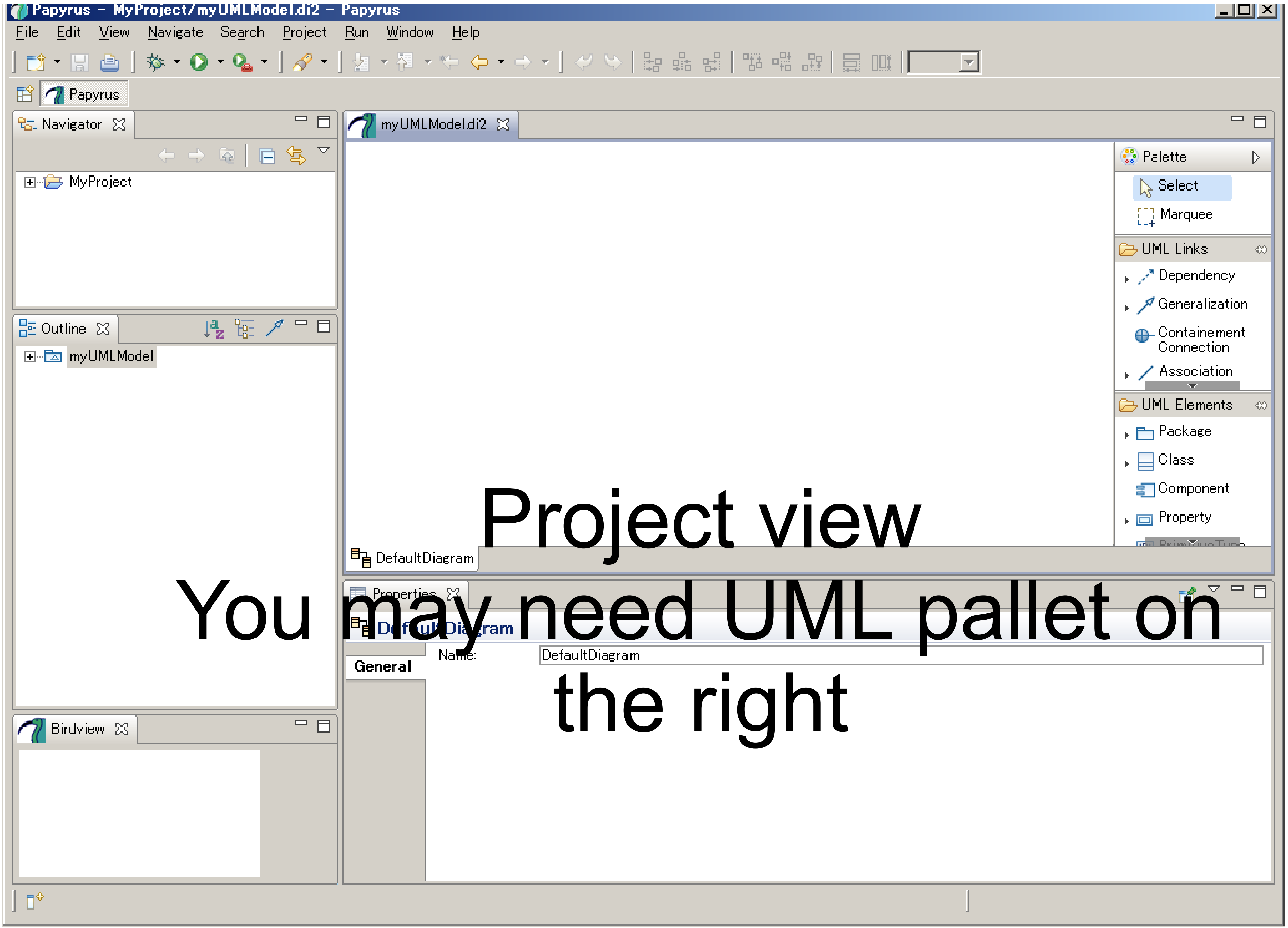
An outline is not available.

An overview is not available.

# Start Papyrus UML

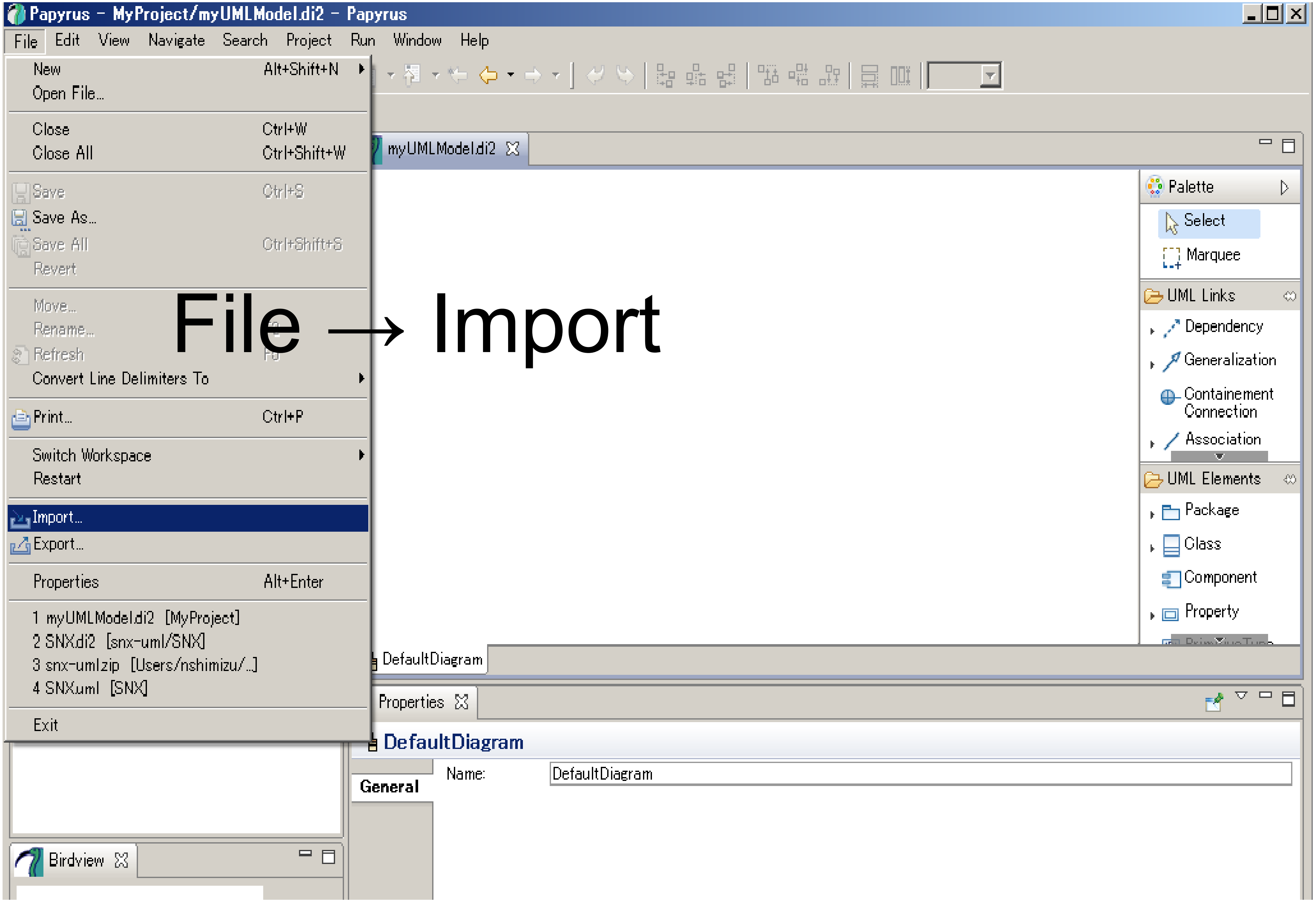
## Create a new Papyrus project





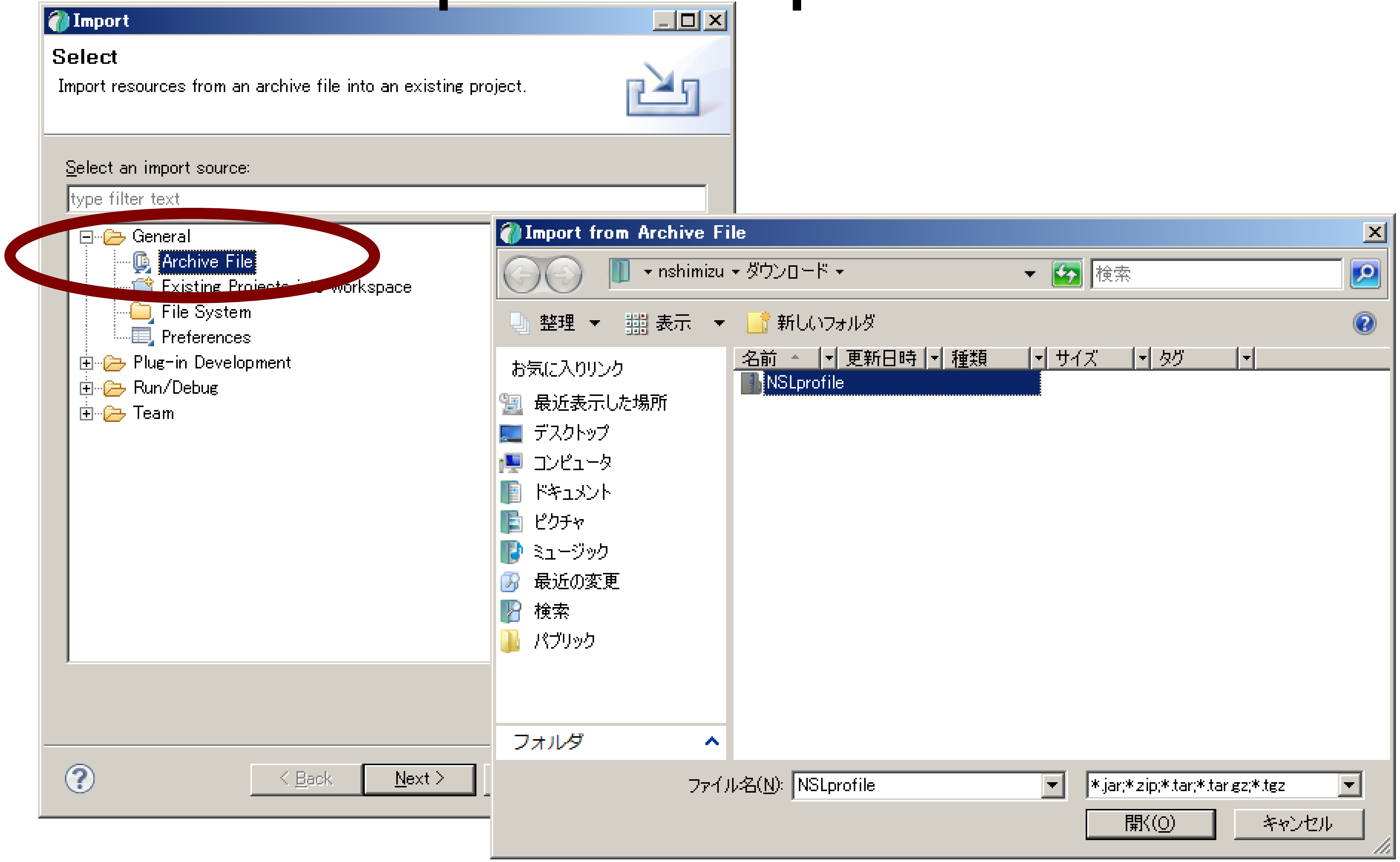
Project view

You may need UML pallet on the right


Papyrus - MyProject/myUMLModel.di2 - Papyrus
File
Edit
View
Navigate
Search
Project
Run
Window
Help
New Alt+Shift+N
Open File...
Close Ctrl+W
Close All Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename...
Refresh
Convert Line Delimiters To
Print... Ctrl+P
Switch Workspace
Restart
Import...
Export...
Properties Alt+Enter
1 myUMLModel.di2 [MyProject]
2 SNX.di2 [snx-uml/SNX]
3 snx-uml.zip [Users/nshimizu/...]
4 SNX.uml [SNX]
Exit
myUMLModel.di2
Palette
Select
Marquee
UML Links
Dependency
Generalization
Containment Connection
Association
UML Elements
Package
Class
Component
Property
DefaultDiagram
Properties
DefaultDiagram
General
Name:
DefaultDiagram
Birdview

File → Import

# Select Archive File NSLprofile.zip



Import

Select

Import resources from an archive file into an existing project.

Select an import source:

type filter text

- General
  - Archive File
  - Existing Projects into Workspace
  - File System
  - Preferences
- Plug-in Development
- Run/Debug
- Team

< Back  Next >

Import from Archive File

nshimizu ▸ ダウンロード ▸

検索

整理 ▾  表示 ▾  新しいフォルダ

お気に入りリンク

- 最近表示した場所
- デスクトップ
- コンピュータ
- ドキュメント
- ピクチャ
- ミュージック
- 最近の変更
- 検索
- パブリック

フォルダ

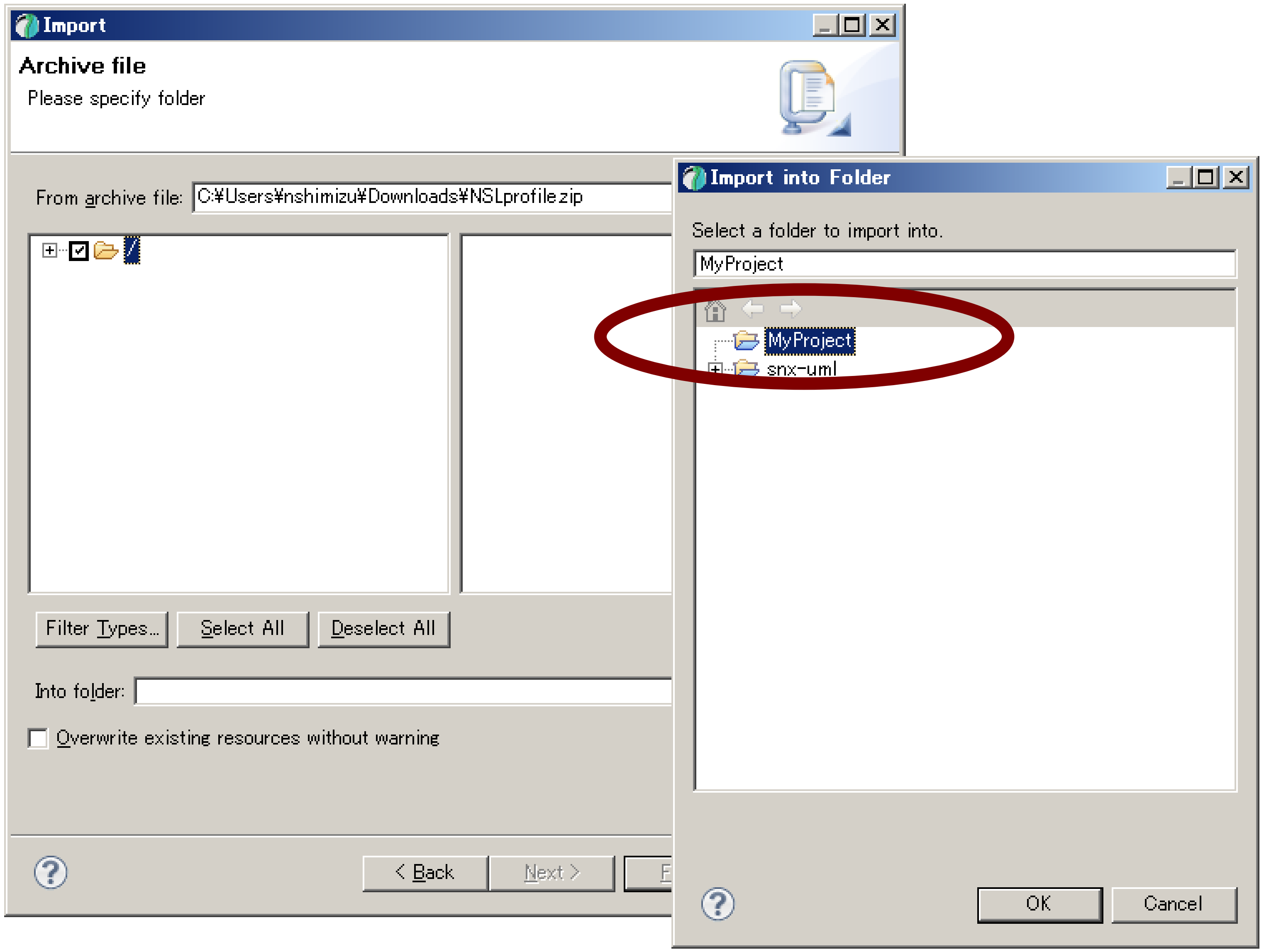
| 名前 | 更新日時 | 種類 | サイズ | タグ |
|---|---|---|---|---|
| NSLprofile | | | | |

ファイル名(N): NSLprofile

*.jar;*.zip;*.tar;*.tar.gz;*.tgz

開く(O)  キャンセル

# Import into current project



Import

Archive file

Please specify folder

From archive file: C:¥Users¥nshimizu¥Downloads¥NSLprofile.zip

Into folder:

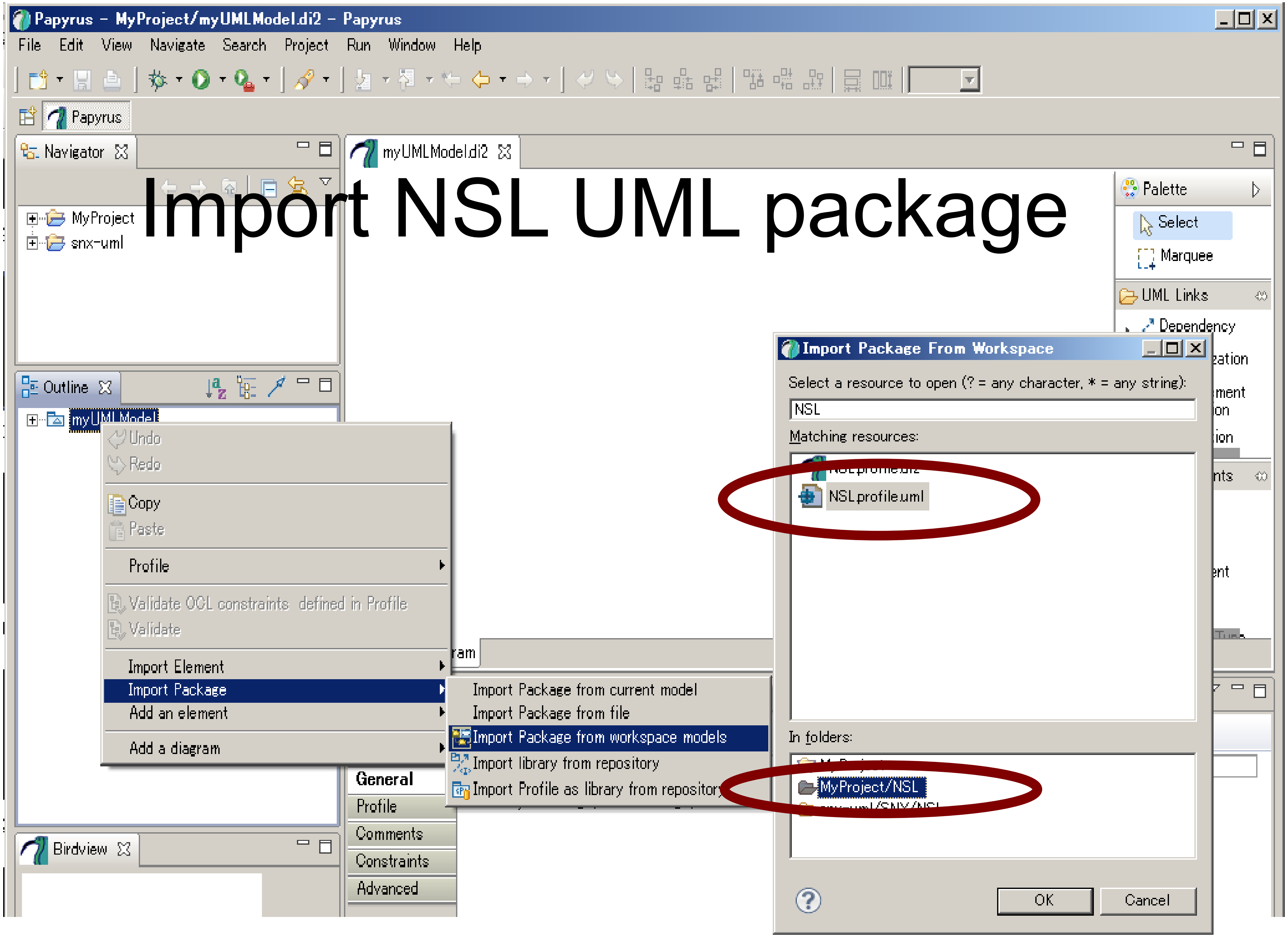
☐ Overwrite existing resources without warning

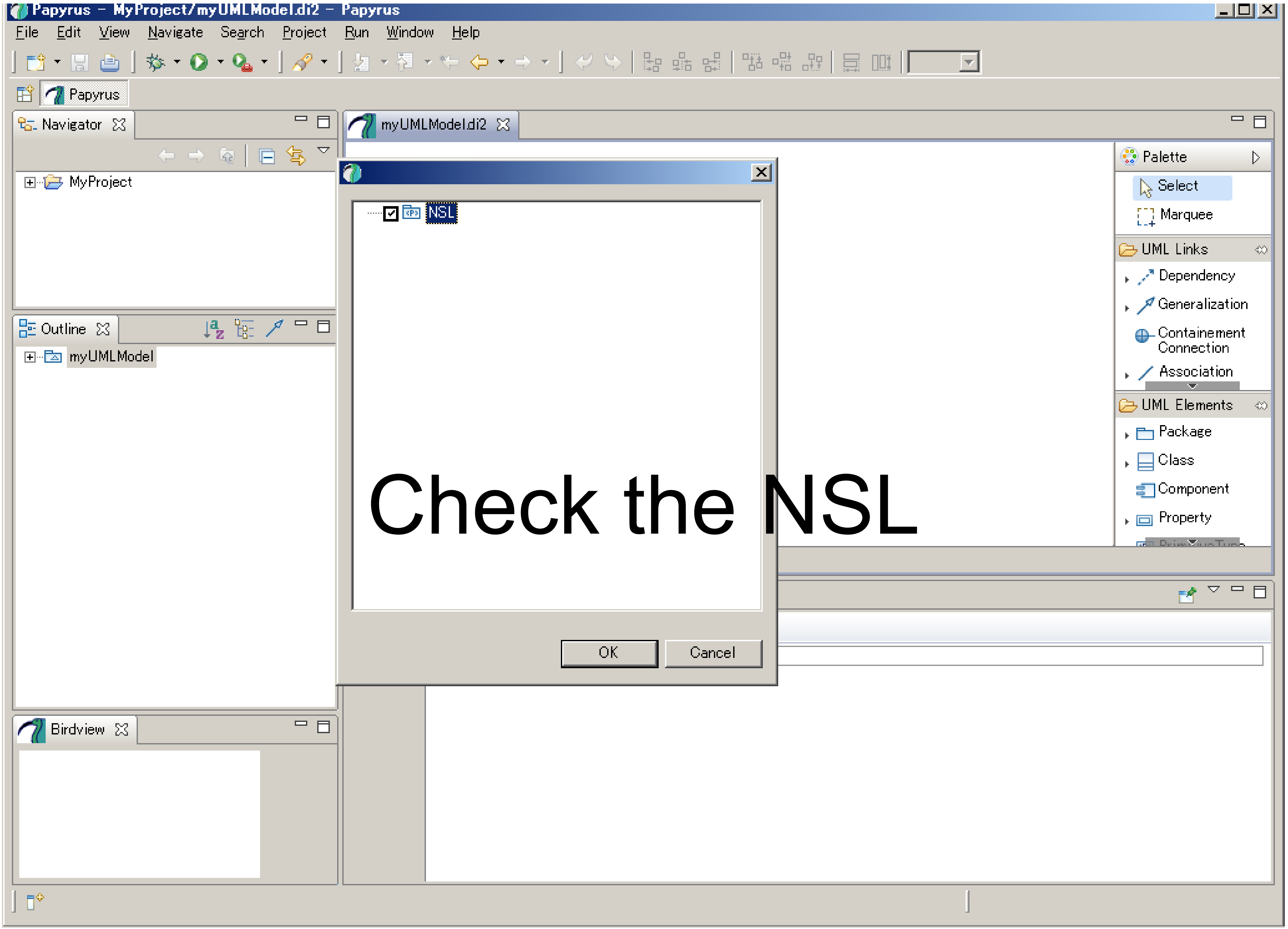
Import into Folder

Select a folder to import into.

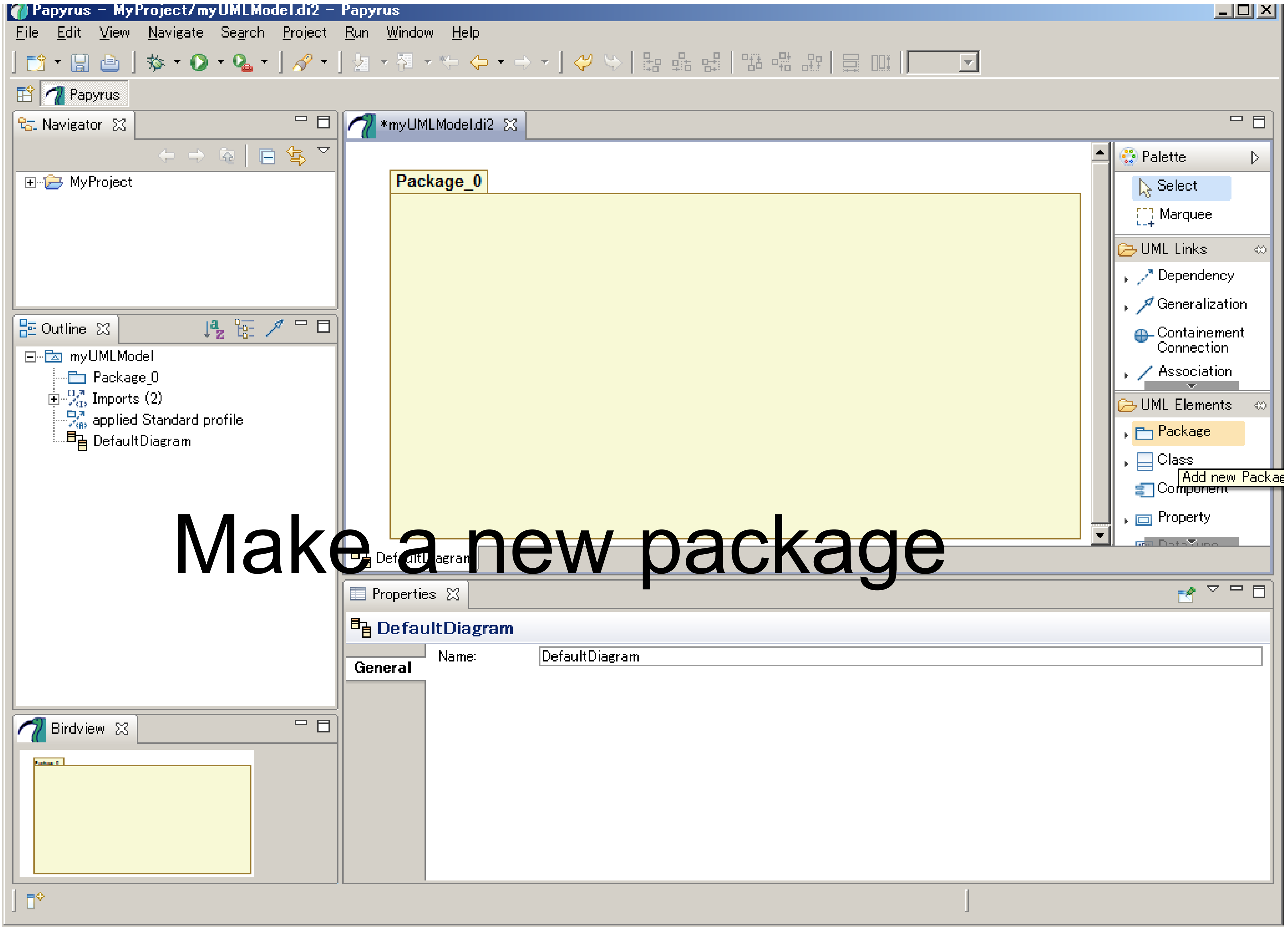
MyProject

# Import NSL UML package


Papyrus - MyProject/myUMLModel.di2 - Papyrus
File Edit View Navigate Search Project Run Window Help
Navigator
MyProject
snx-uml
Outline
myUMLModel
Undo
Redo
Copy
Paste
Profile
Validate OCL constraints defined in Profile
Validate
Import Element
Import Package
Add an element
Add a diagram
Import Package from current model
Import Package from file
Import Package from workspace models
Import library from repository
Import Profile as library from repository
General
Profile
Comments
Constraints
Advanced
Birdview
Palette
Select
Marquee
UML Links
Dependency
Import Package From Workspace
Select a resource to open (? = any character, * = any string):
NSL
Matching resources:
NSLprofile.di2
NSLprofile.uml
In folders:
MyProject/NSL
OK
Cancel



Check the NSL

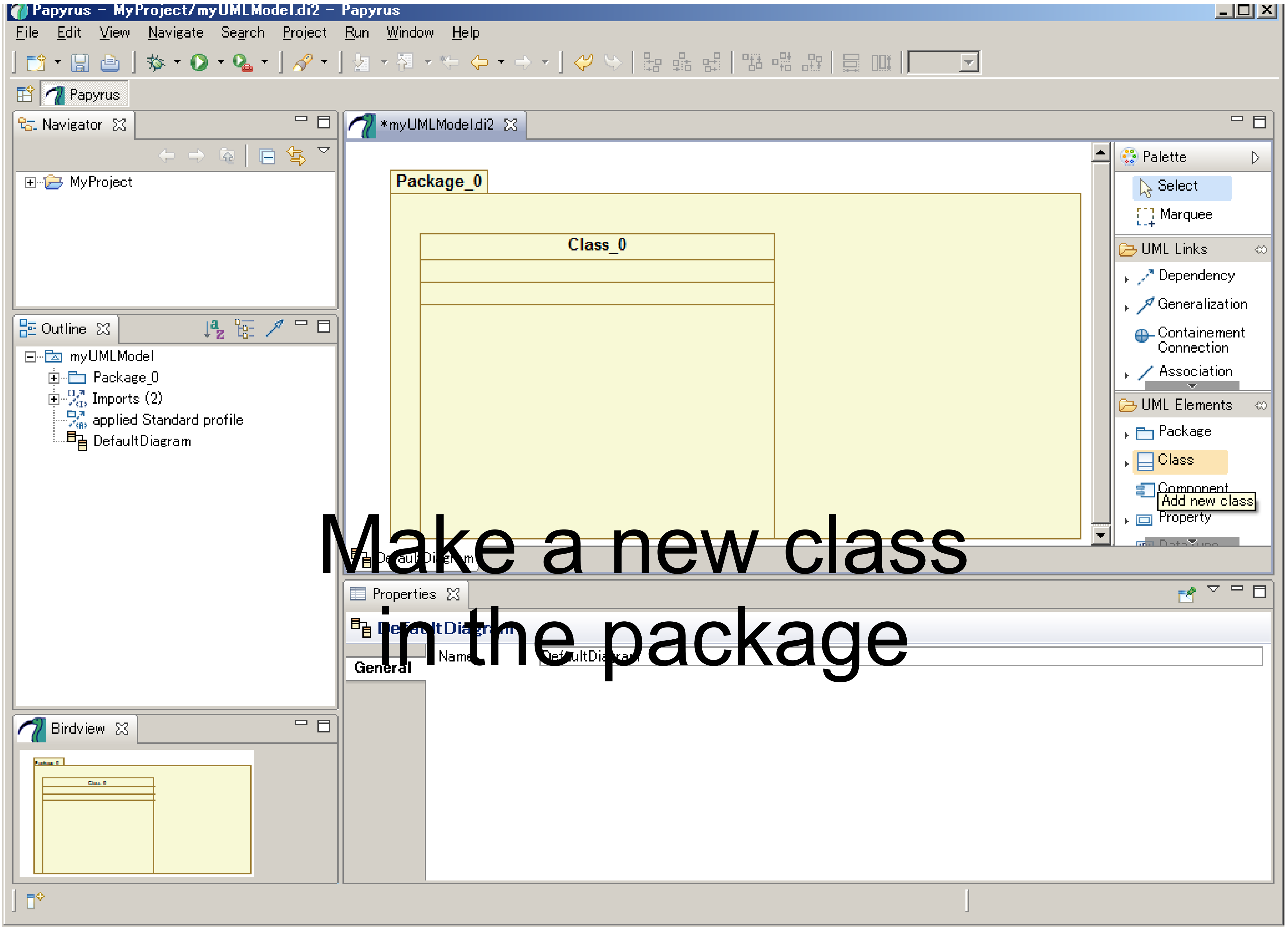


DefaultDiagram

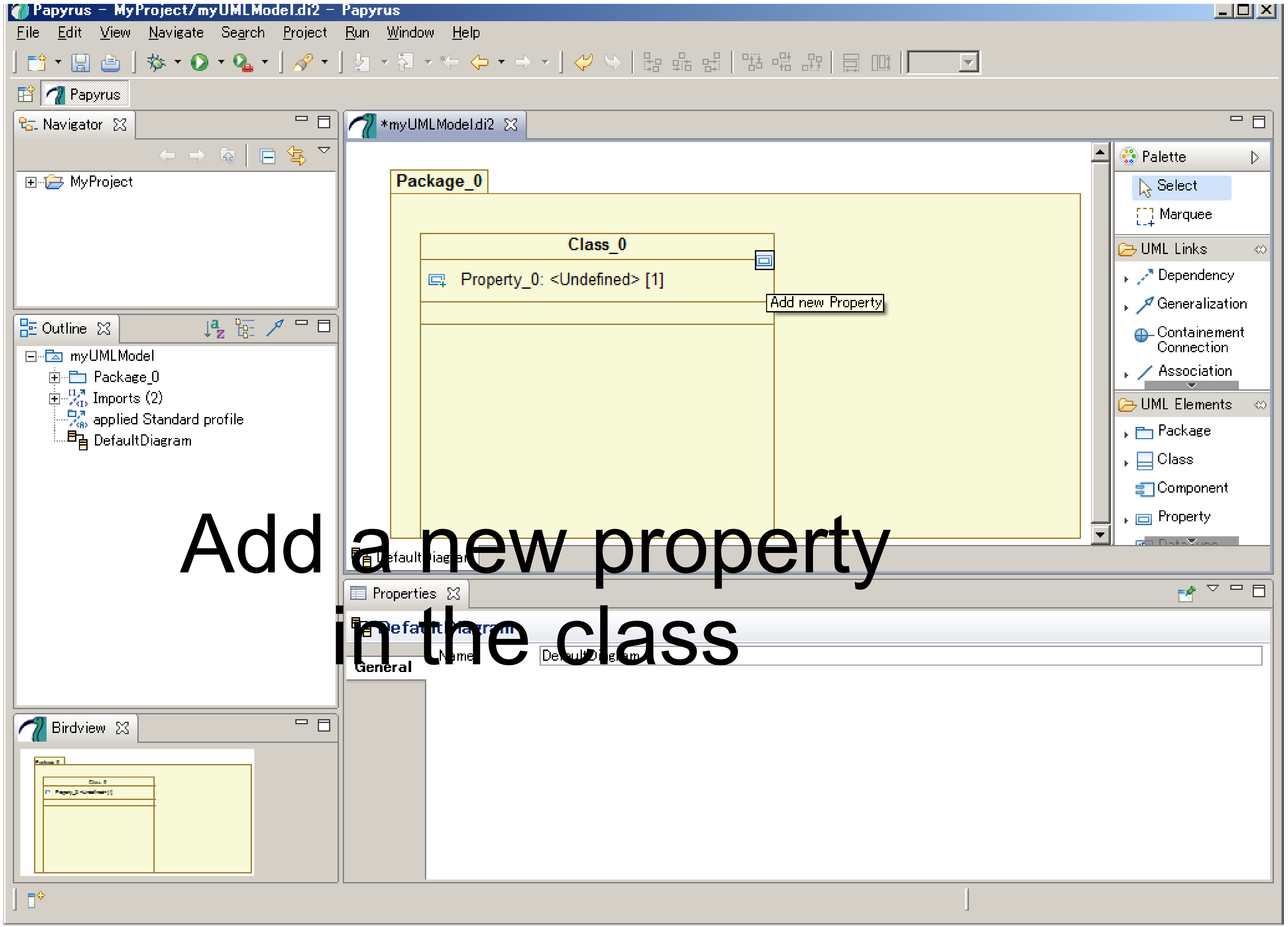
General

Name: DefaultDiagram

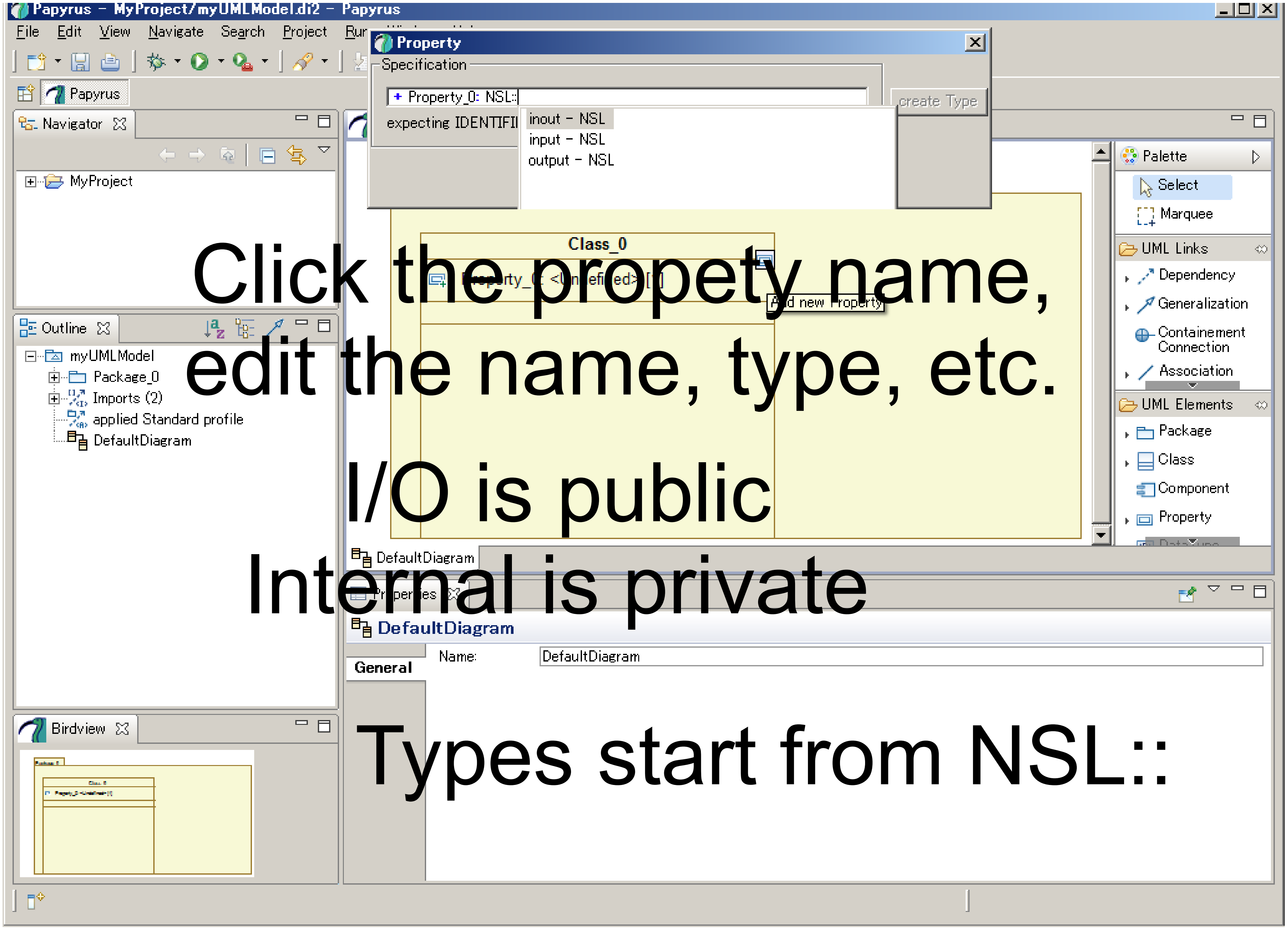
Make a new package



Make a new class in the package



Add a new property in the class



Property

Specification

+ Property_0: NSL::

expecting IDENTIFI

inout - NSL
input - NSL
output - NSL

Class_0

Property_0: <Undefined> [1]

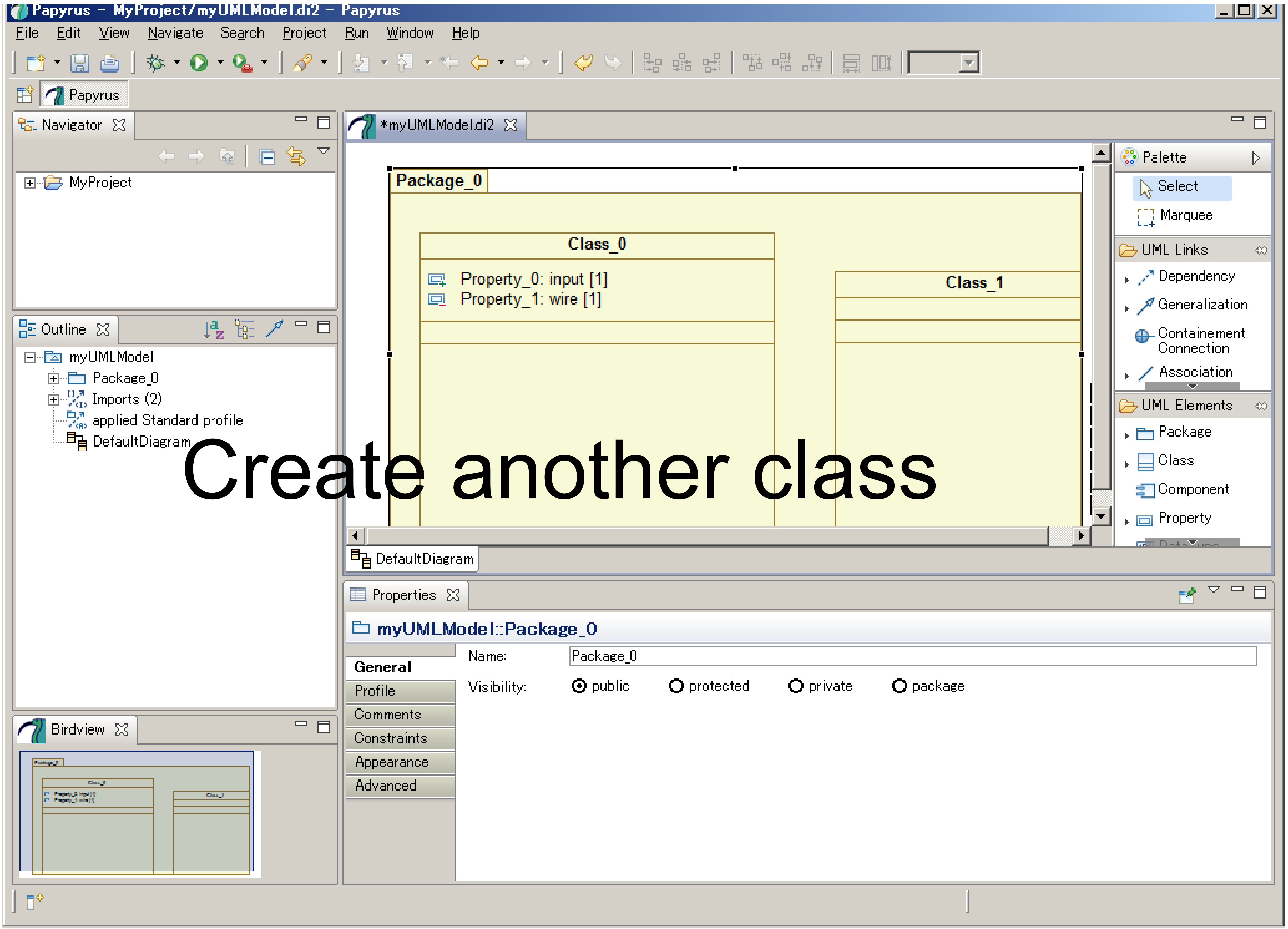
Name: DefaultDiagram

Click the propety name, edit the name, type, etc.

I/O is public

Internal is private

Types start from NSL::



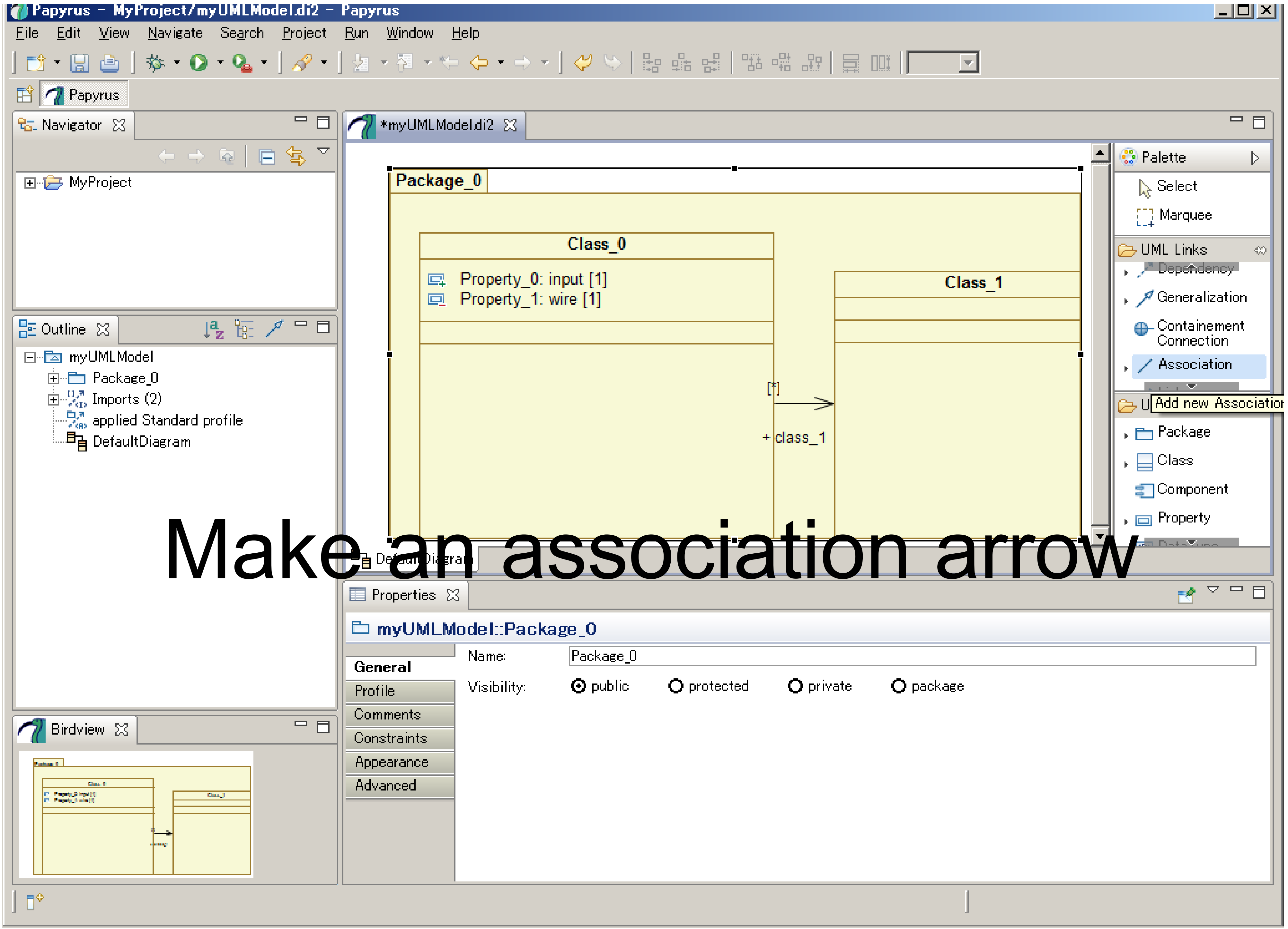
myUMLModel::Package_0

General | Profile | Comments | Constraints | Appearance | Advanced

Name: Package_0

Visibility: [✓ public] protected private package

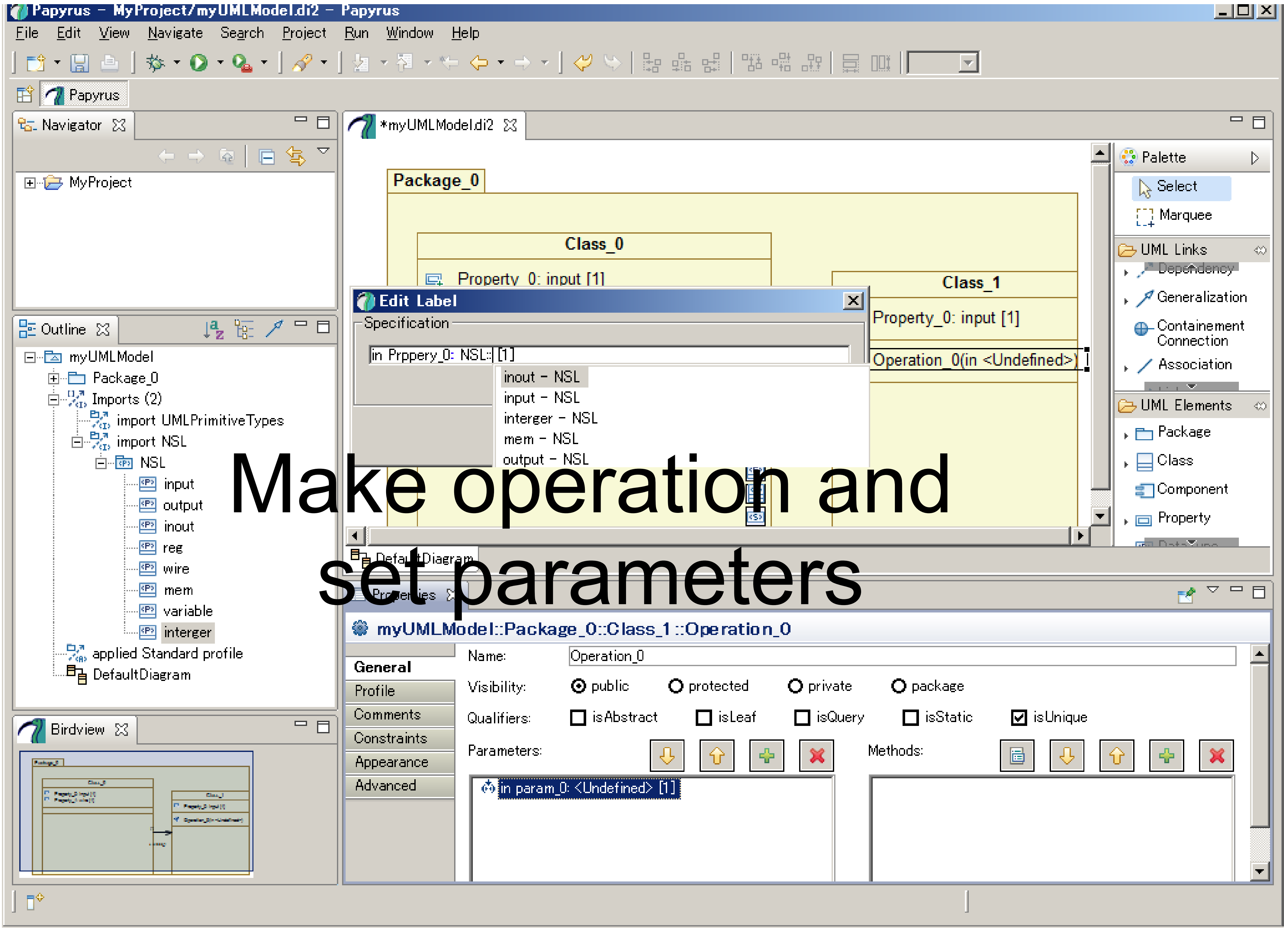
# Create another class



Name: Package_0

Visibility: [✓ public] protected private package

# Make an association arrow

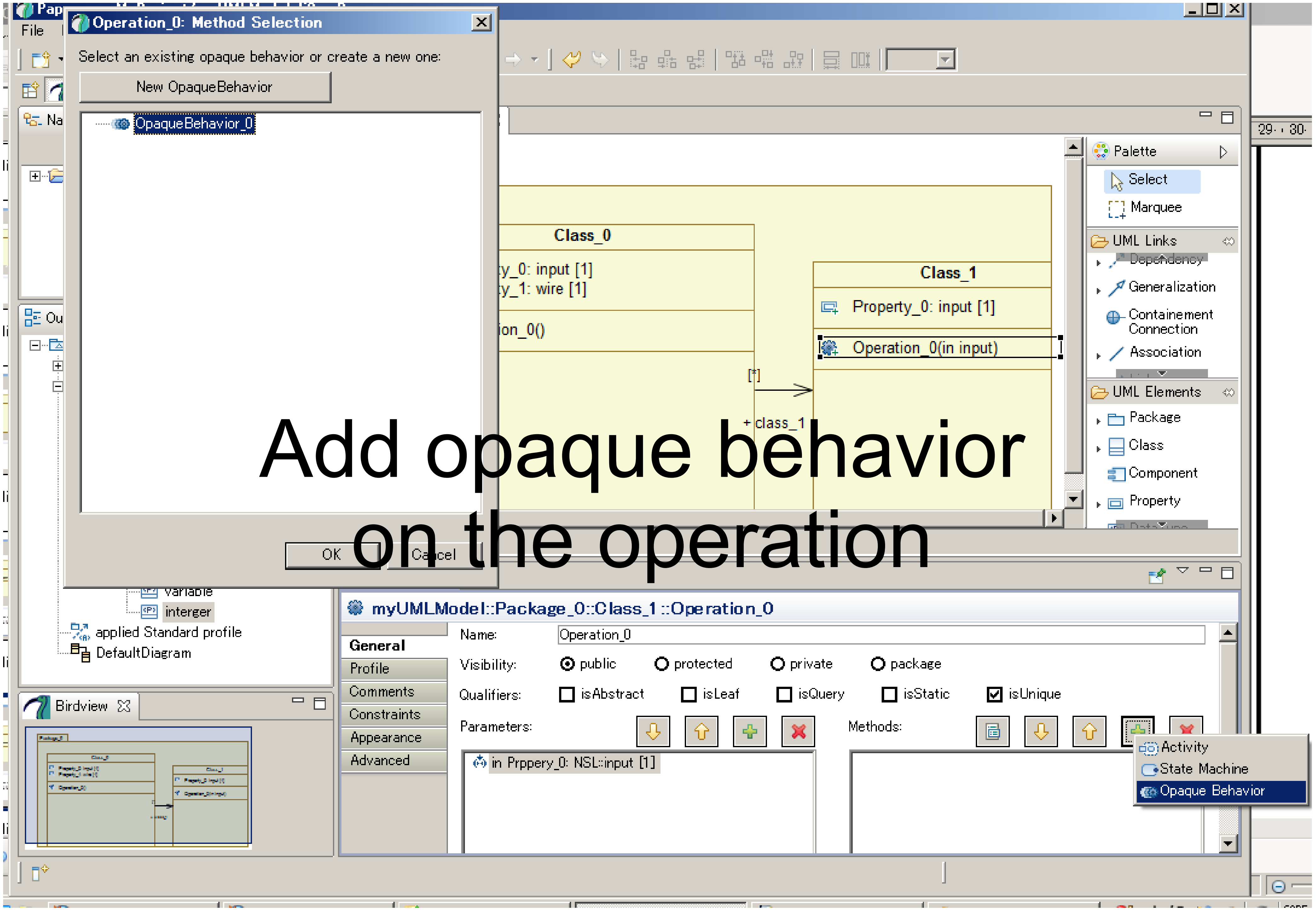


Name: Operation_0

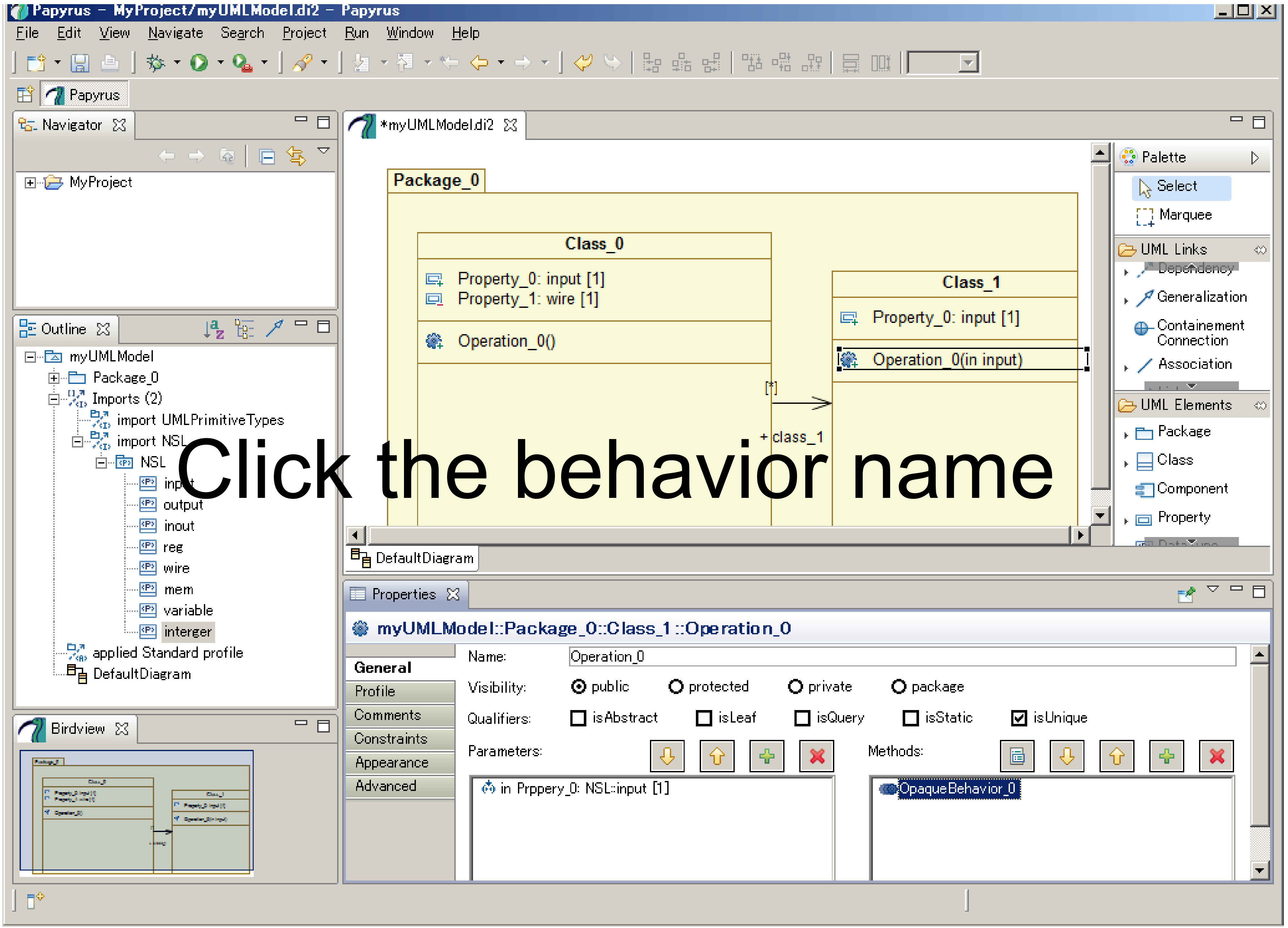
Visibility: [✓ public] protected private package

Qualifiers: isAbstract isLeaf isQuery isStatic [✓ isUnique]

# Make operation and set parameters



Add opaque behavior on the operation



myUMLModel::Package_0::Class_1::Operation_0

Name: Operation_0

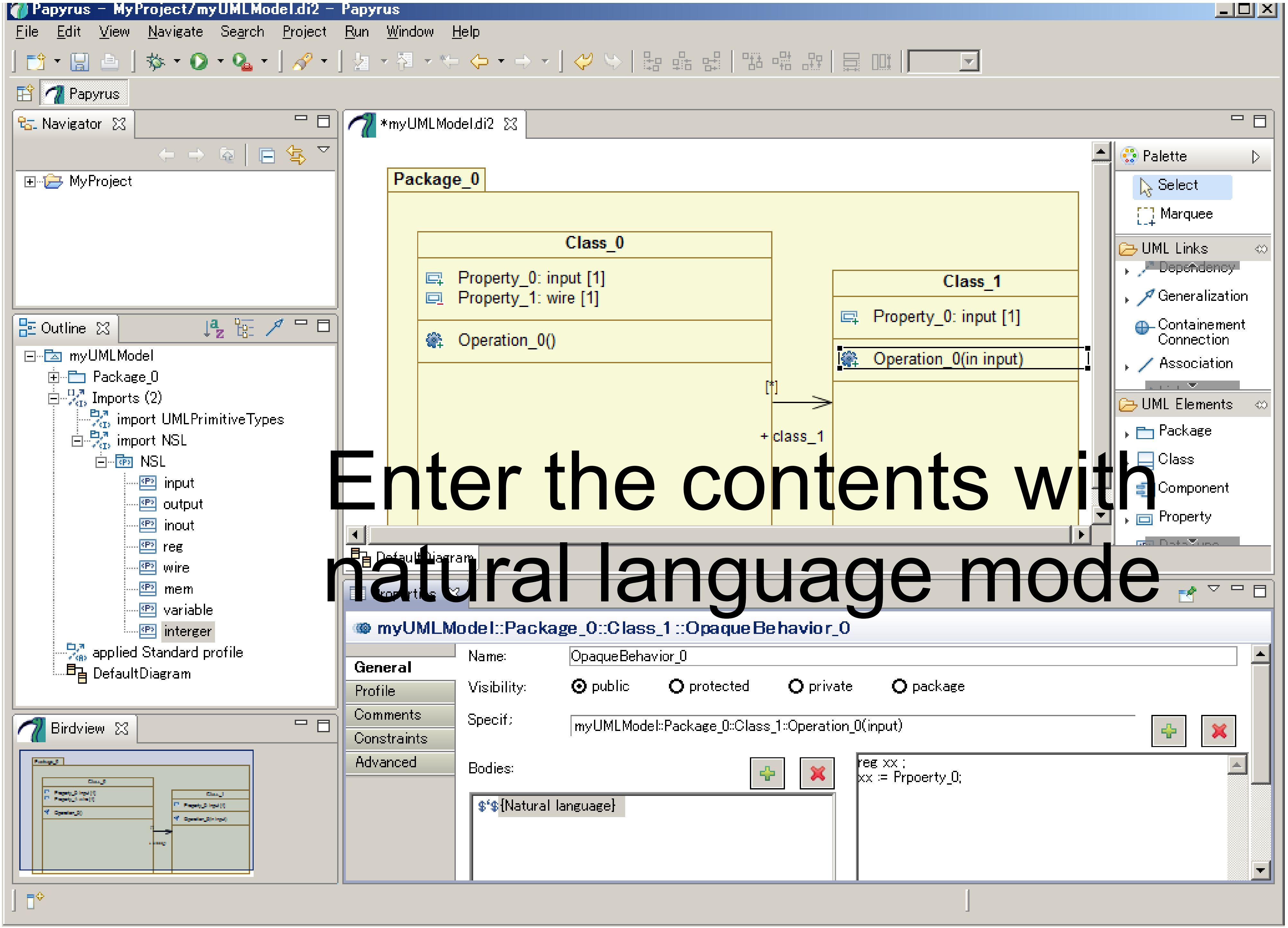
Visibility: [✓ public] protected private package

Qualifiers: isAbstract isLeaf isQuery isStatic [✓ isUnique]

Parameters: in Prppery_0: NSL::input [1]

Methods: OpaqueBehavior_0

Click the behavior name



Enter the contents with natural language mode

myUMLModel::Package_0::Class_1::OpaqueBehavior_0

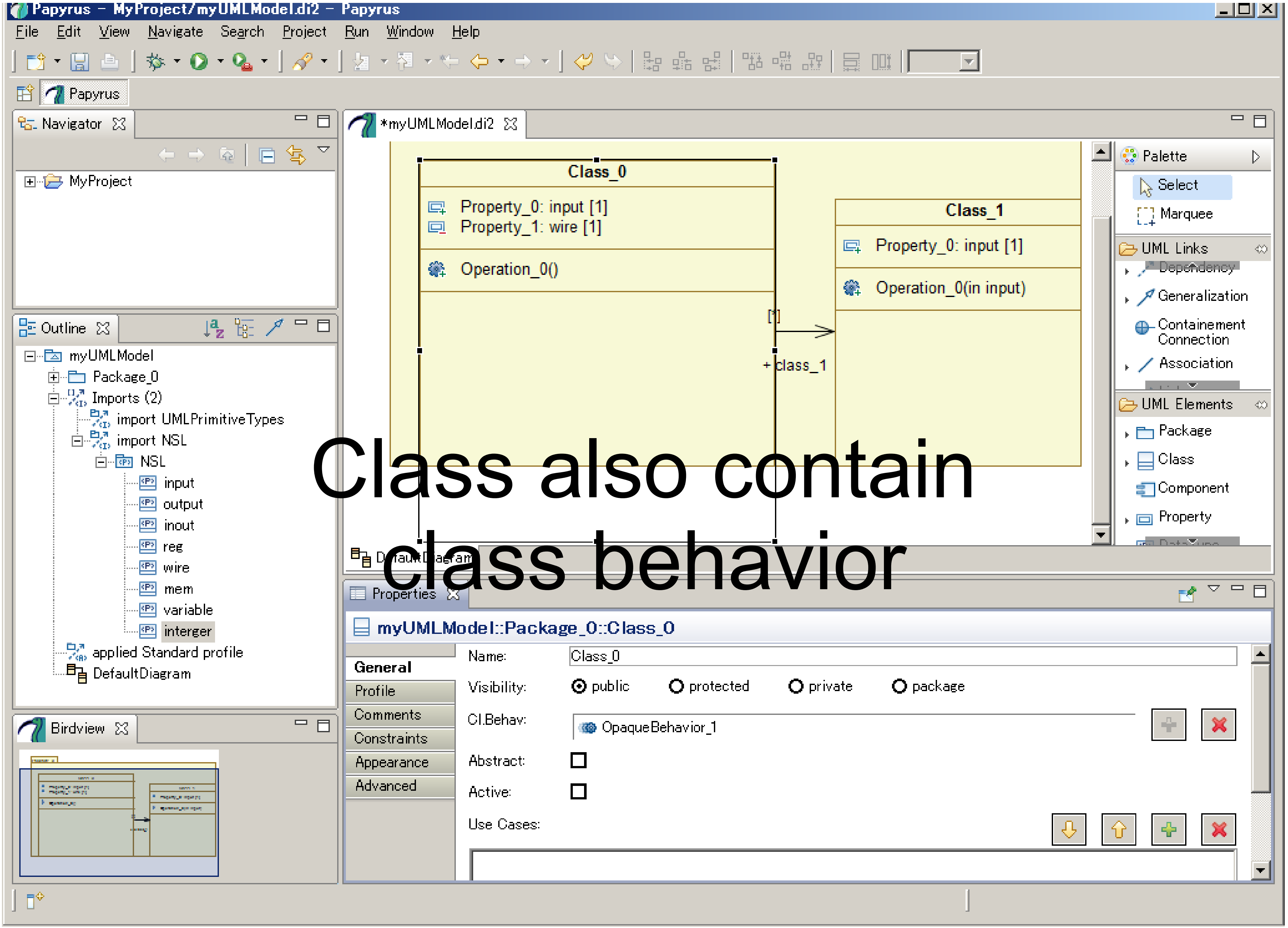
Name: OpaqueBehavior_0

Visibility: [✓ public] protected private package

Specif: myUMLModel::Package_0::Class_1::Operation_0(input)

Bodies: $'${Natural language}

```
reg xx ;
xx := Prpoerty_0;
```



myUMLModel::Package_0::Class_0

Name: Class_0

Visibility: [✓ public] protected private package

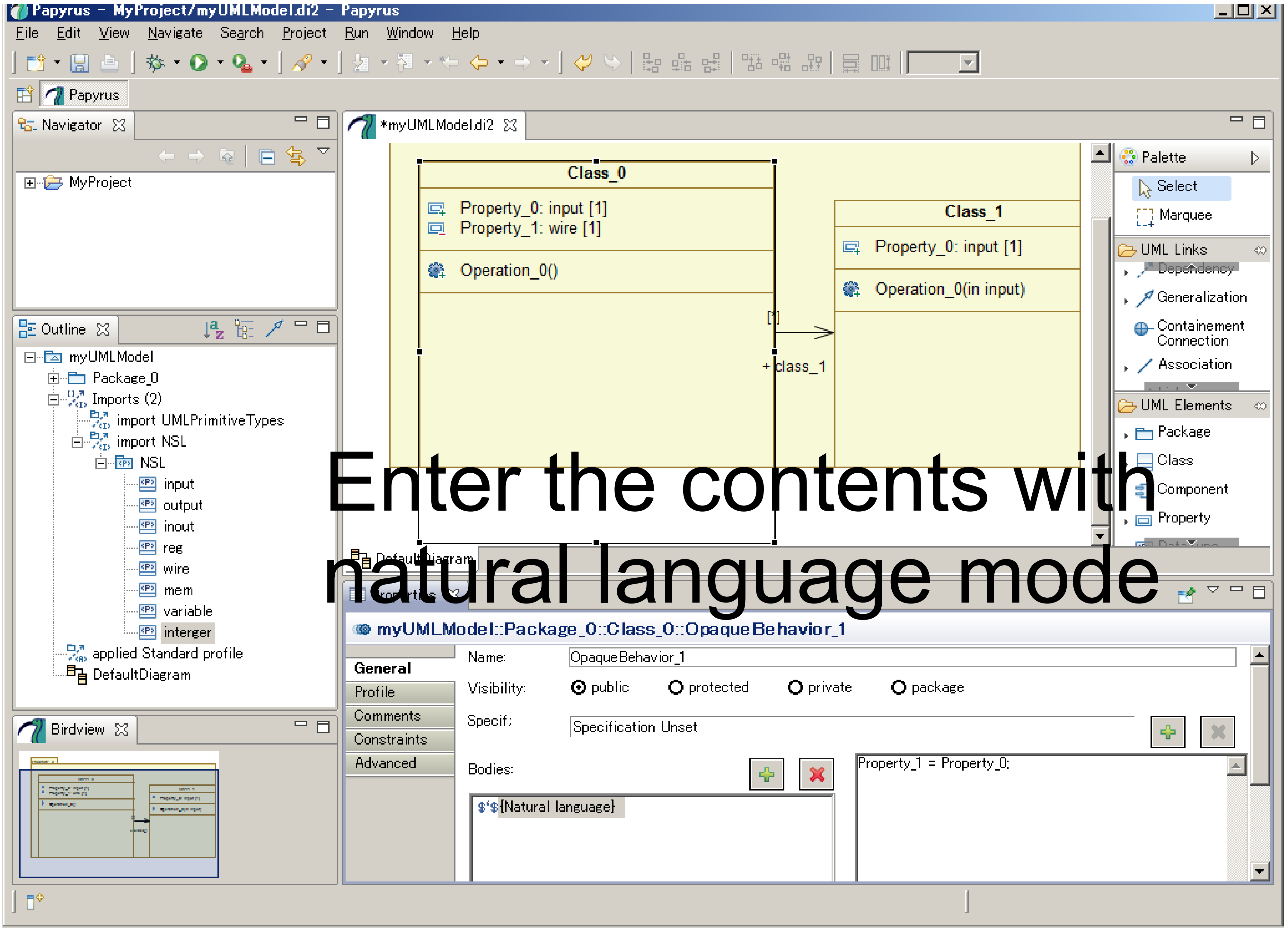
Cl.Behav: OpaqueBehavior_1

Abstract: ☐

Active: ☐

Use Cases:

Class also contain class behavior


Papyrus - MyProject/myUMLModel.di2 - Papyrus

Enter the contents with natural language mode

myUMLModel::Package_0::Class_0::OpaqueBehavior_1

Name: OpaqueBehavior_1

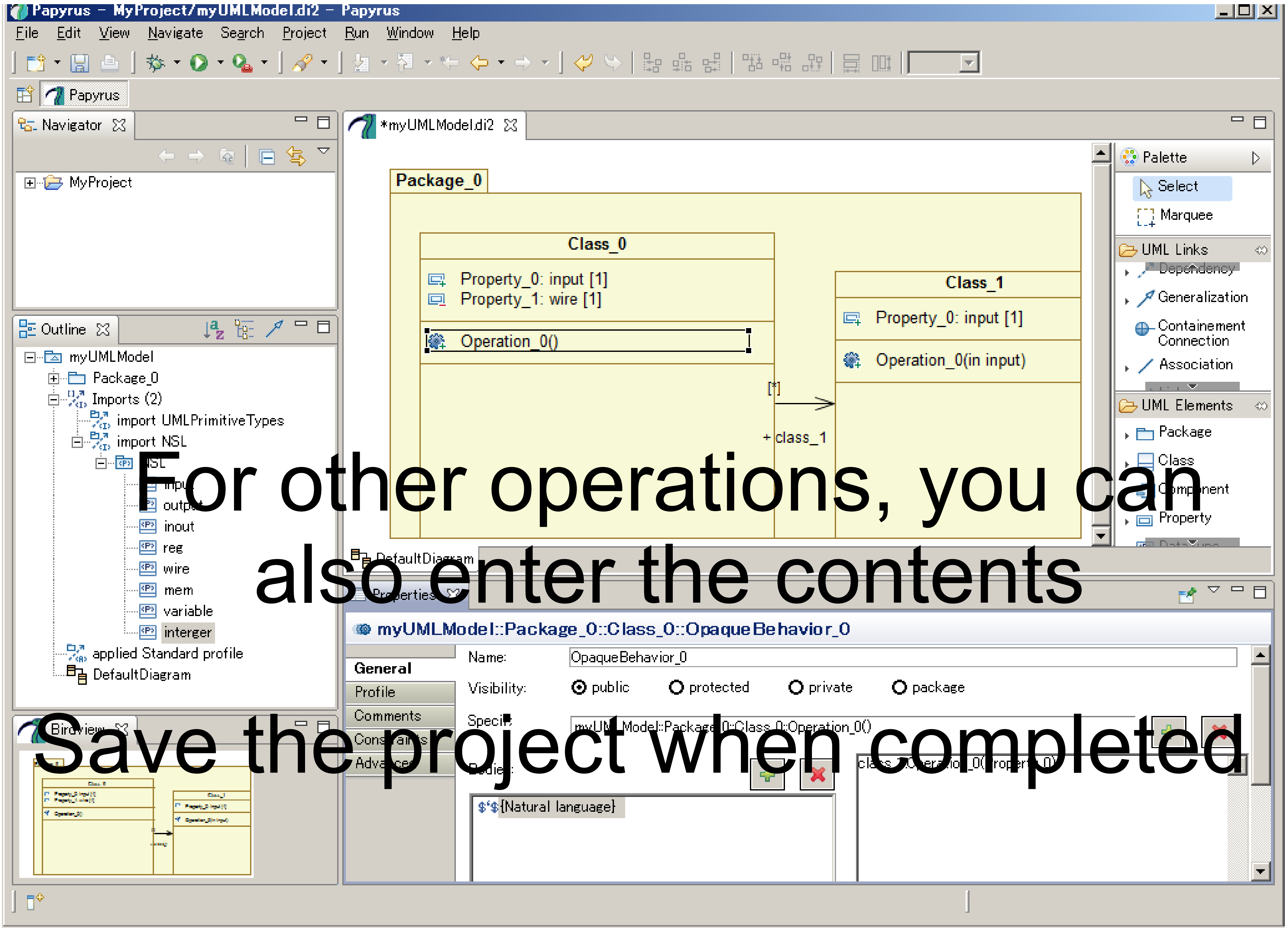
Visibility: [✓ public] protected private package

Specif: Specification Unset

Bodies:

$‘$ {Natural language}

```
Property_1 = Property_0;
```



myUMLModel::Package_0::Class_0::OpaqueBehavior_0

Name: OpaqueBehavior_0

Visibility: [✓ public] protected private package

Specification: myUMLModel::Package_0::Class_0::Operation_0()

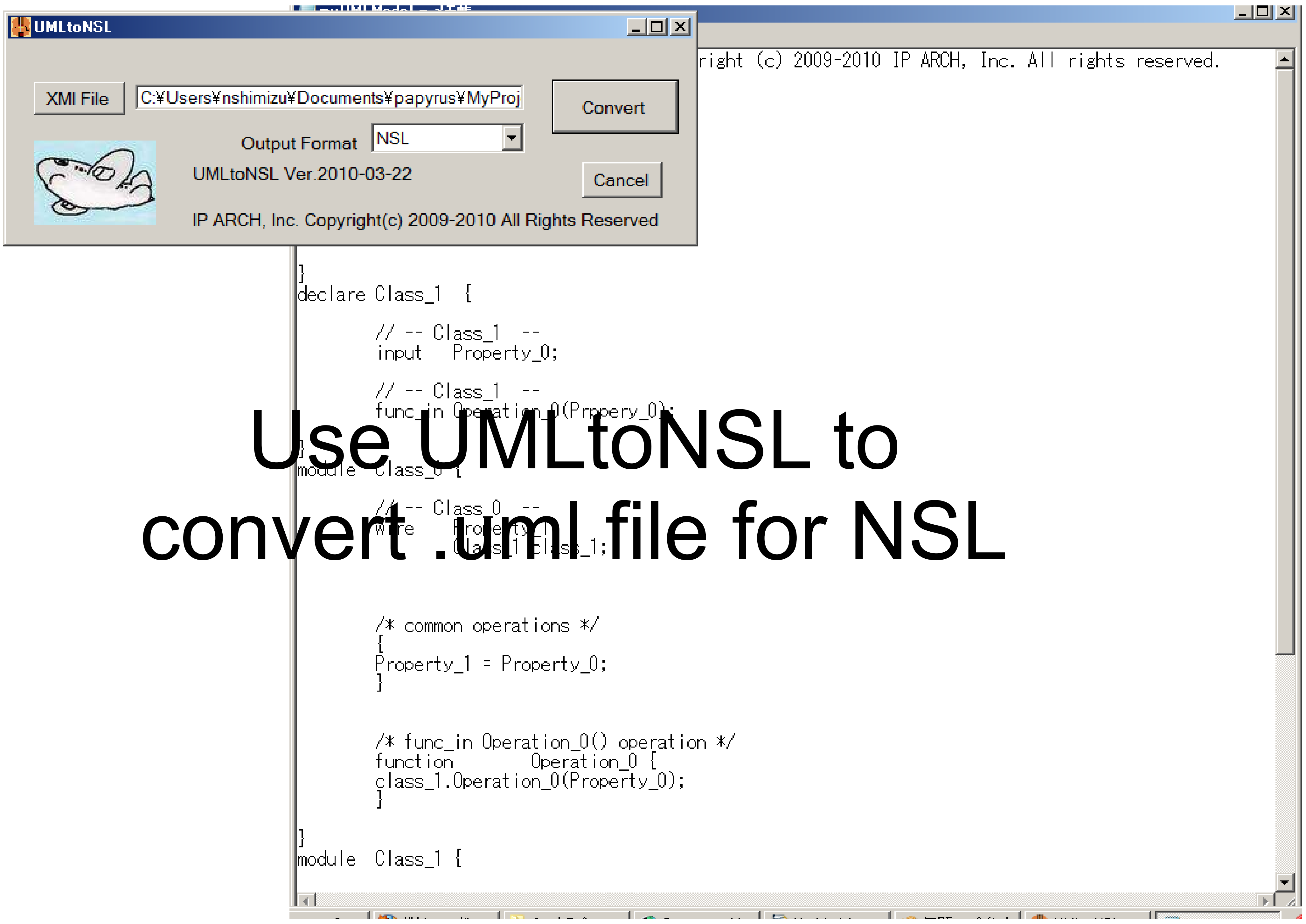
Bodies: class_1.Operation_0(property_0)

{Natural language}

For other operations, you can also enter the contents

Save the project when completed

# Use UMLtoNSL to convert .uml file for NSL



C:¥Users¥nshimizu¥Documents¥papyrus¥MyProj

Output Format NSL

UMLtoNSL Ver.2010-03-22

IP ARCH, Inc. Copyright(c) 2009-2010 All Rights Reserved

```
right (c) 2009-2010 IP ARCH, Inc. All rights reserved.

}
declare Class_1  {

        // -- Class_1  --
        input   Property_0;

        // -- Class_1  --
        func_in Operation_0(Prppery_0);
}
module  Class_0 {
        // -- Class_0  --
        wire    Property_1
        Class_1 Class_1;


        /* common operations */
        {
        Property_1 = Property_0;
        }


        /* func_in Operation_0() operation */
        function        Operation_0 {
        class_1.Operation_0(Property_0);
        }

}
module  Class_1 {
```