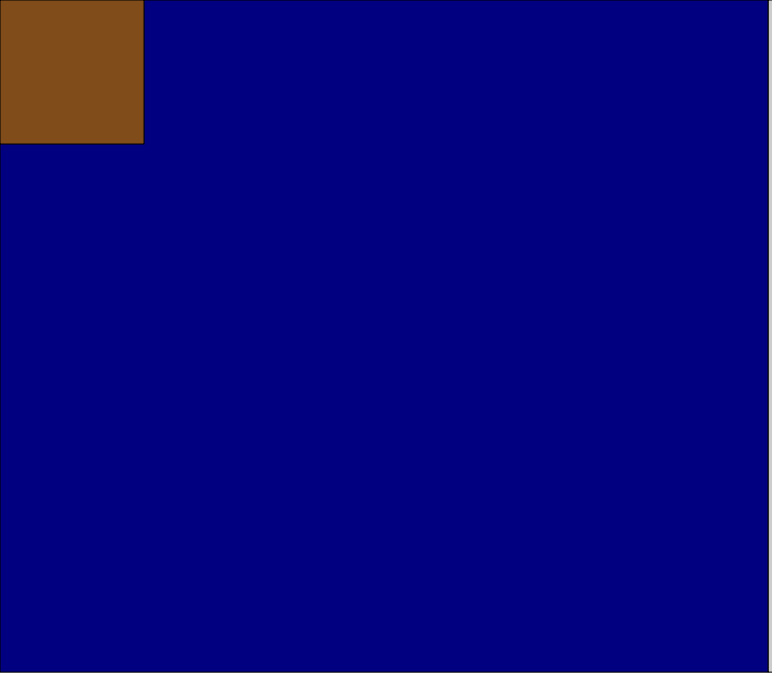
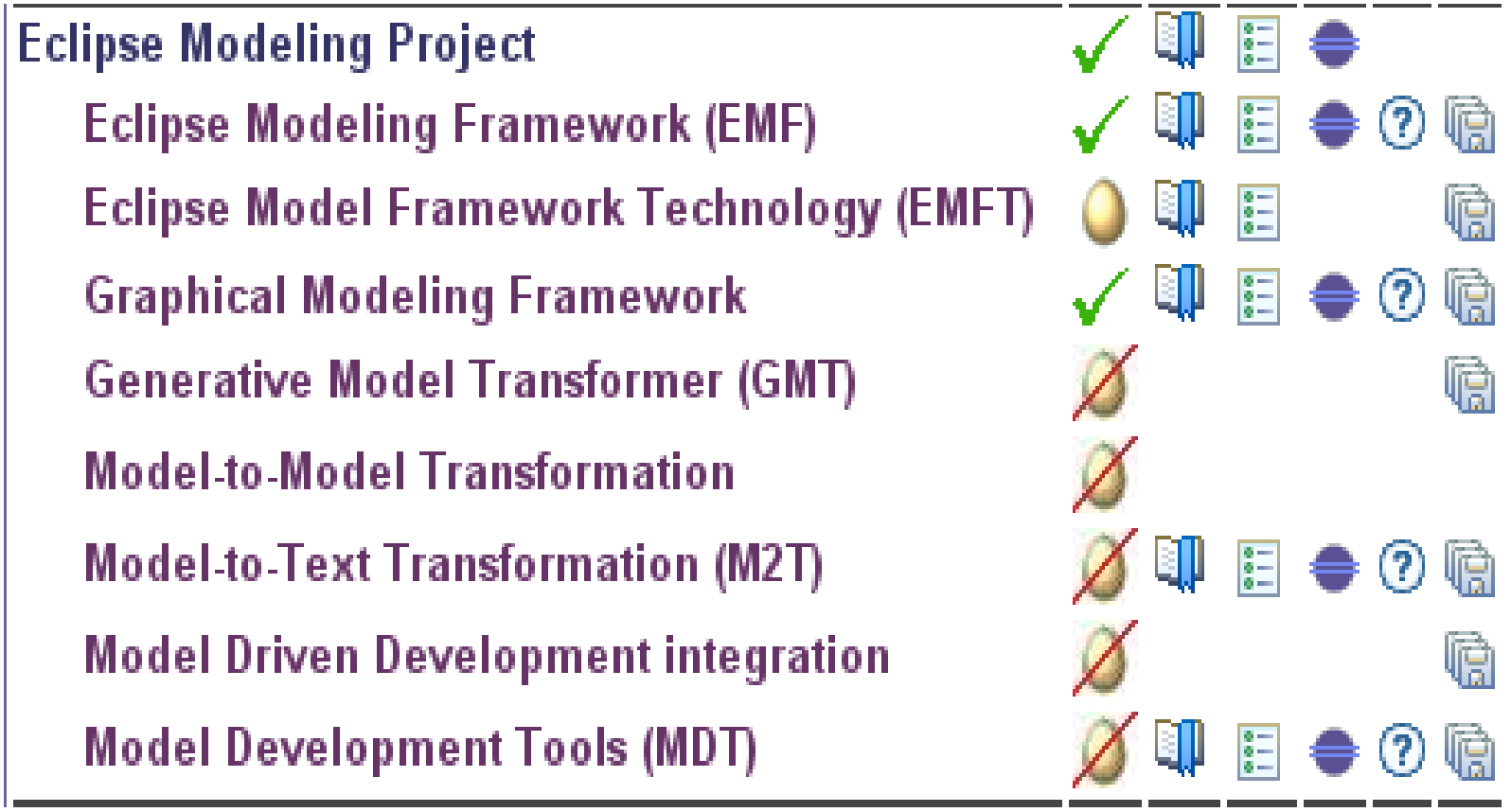


EMF @ Tues

EMF

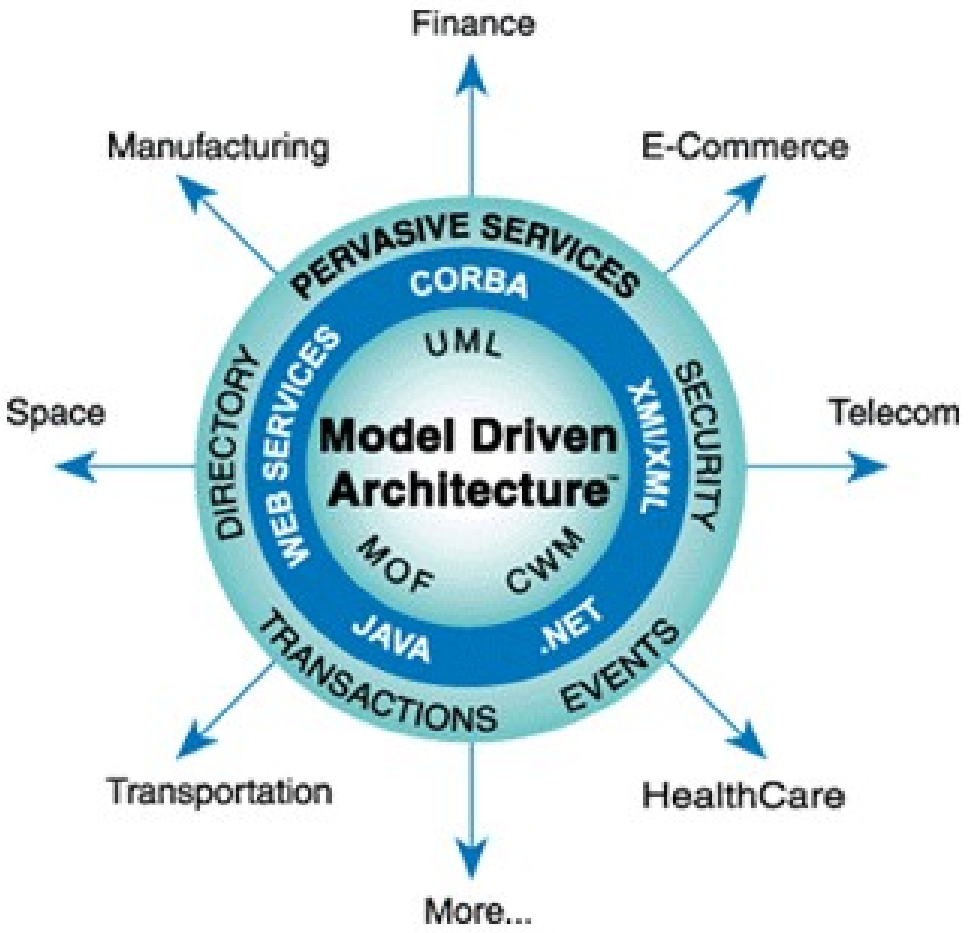

Eclipse Modeling Project
Eclipse Modeling Framework (EMF)
Eclipse Model Framework Technology (EMFT)
Graphical Modeling Framework
Generative Model Transformer (GMT)
Model-to-Model Transformation
Model-to-Text Transformation (M2T)
Model Driven Development integration
Model Development Tools (MDT)

MDA

Model Driven Architecture - http://www.omg.org/mda/

- Архитектура за разработка на софтуер предложена от OMG.
- Използват се отворени стандарти за моделиране
  - UML
  - MOF
  - XMI
  - CWM
- Разработваната система се специфицира като платформено независим модел (PIM). Представлява „метамодел“
- PIM се трансформира към специфична имплементация.
- Разделя се бизнес логиката на приложението от имплементацията за съответната платформа. Описва се функционалността и поведението на системата разделена от реалния код, който я реализира.

Model Driven Architecture - http://www.omg.org/mda/



Eclipse Modeling Framework Project

EMF - http://www.eclipse.org/modeling/emf/

„The EMF project is a modeling framework and code generation facility for building tools and other applications based on a structured data model. From a model specification described in XMI, EMF provides tools and runtime support to produce a set of Java classes for the model, along with a set of adapter classes that enable viewing and command-based editing of the model, and a basic editor.“

www.eclipse.org/modeling/emf/

Eclipse Modeling Framework Project

EMF е проект на Eclipse. Подпроектите на EMF са следните.

- EMF
- EMFT – „The Eclipse Modeling Framework Technologies (EMFT) project was initiated to incubate new technologies that extend or complement EMF“
- Query- „The query component provides capabilities to specify and execute queries against EMF model elements and their contents“
- Transaction - „The transaction component provides a model management layer built on top of EMF for managing EMF resources“
- Validation - „The validation component provides capabilities used to ensure model integrity“
- SDO - „Service Data Objects (SDO) is a framework that simplifies and unifies data application development in a service oriented architecture (SOA)“

Eclipse Modeling Framework

Метамодел

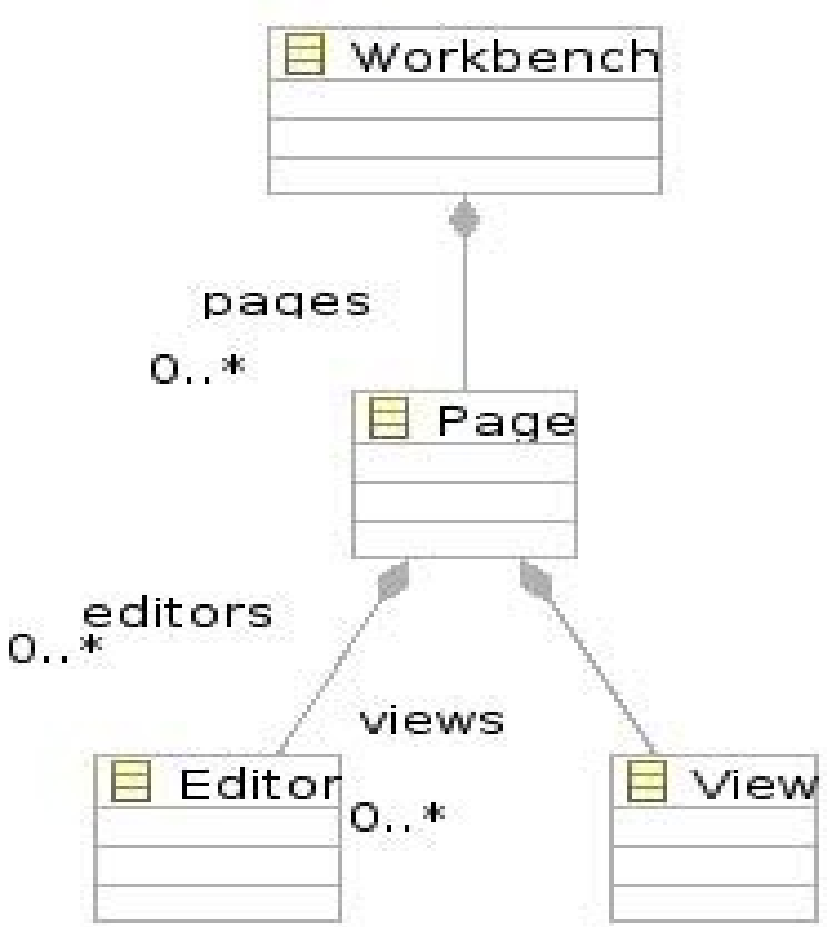
Метамоделът представлява начин да опишем структурата и поведението на системата в платформено независима технология. Представлява по-високо ниво на абстракция, за която може да се генерира част или цялата имплементация.

EMF използва метамодела, за да генерира имплементацията. Генерирания код е на Java. Може да се използва една от следните технологии за създаване на метамодел:

- **Ecore (Meta) Model**
- **XMI**
- **Java Annotations**
- **UML**
- **XML Schema**
- **и др**

Eclipse Modeling Framework - Пример

Пример:
1. Да се изгради метамодел(ecore) описващ следните обекти:



Eclipse Modeling Framework

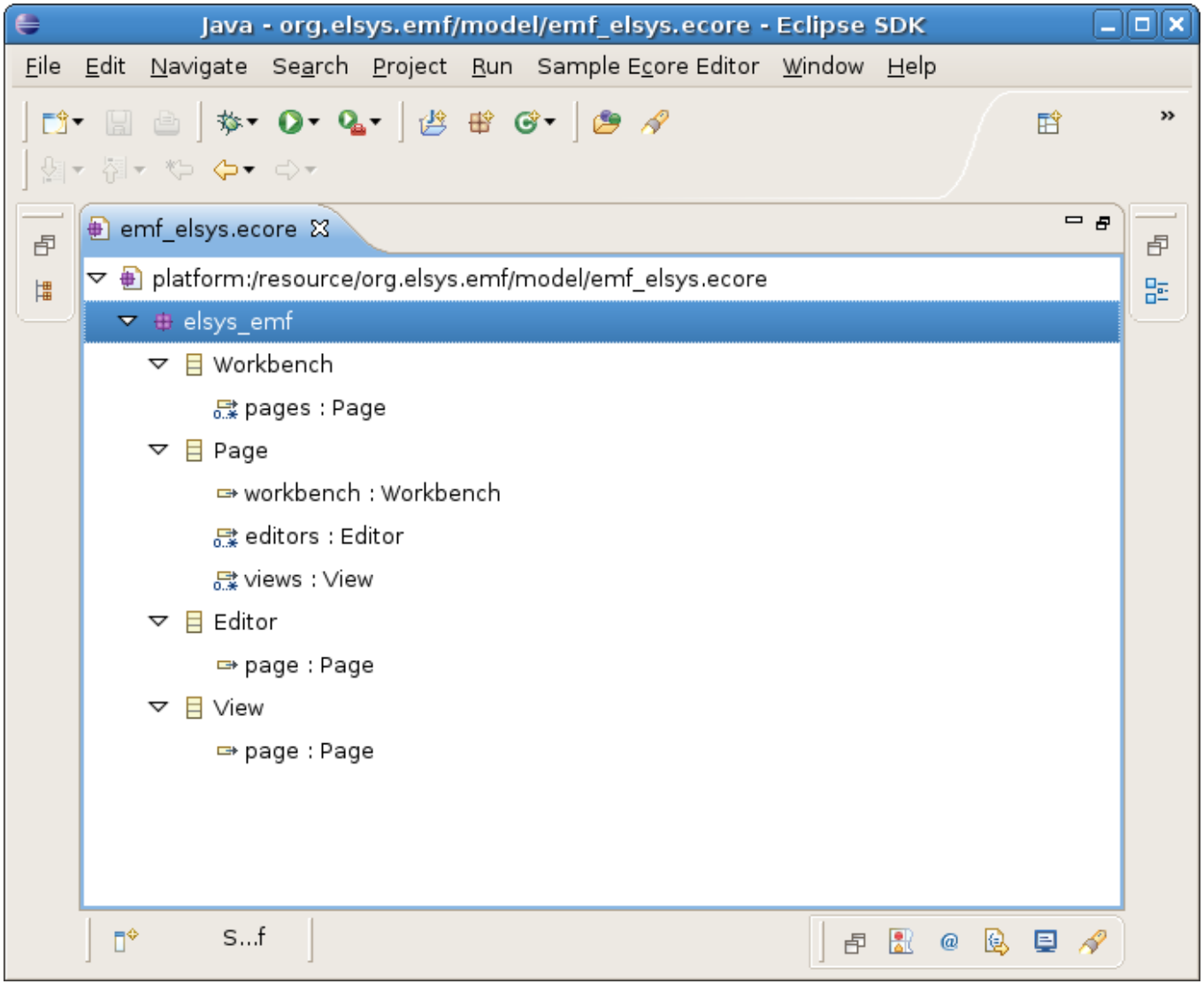
2. Да се създаде междинен метамодел описващ генерацията – genmodel

3. Да се генерира Java кода имплементиращ функционалността описана чрез ecore файла.

4. Да се генерира визуален редактор позволяващ редакцията и запаметяването на описания модел.

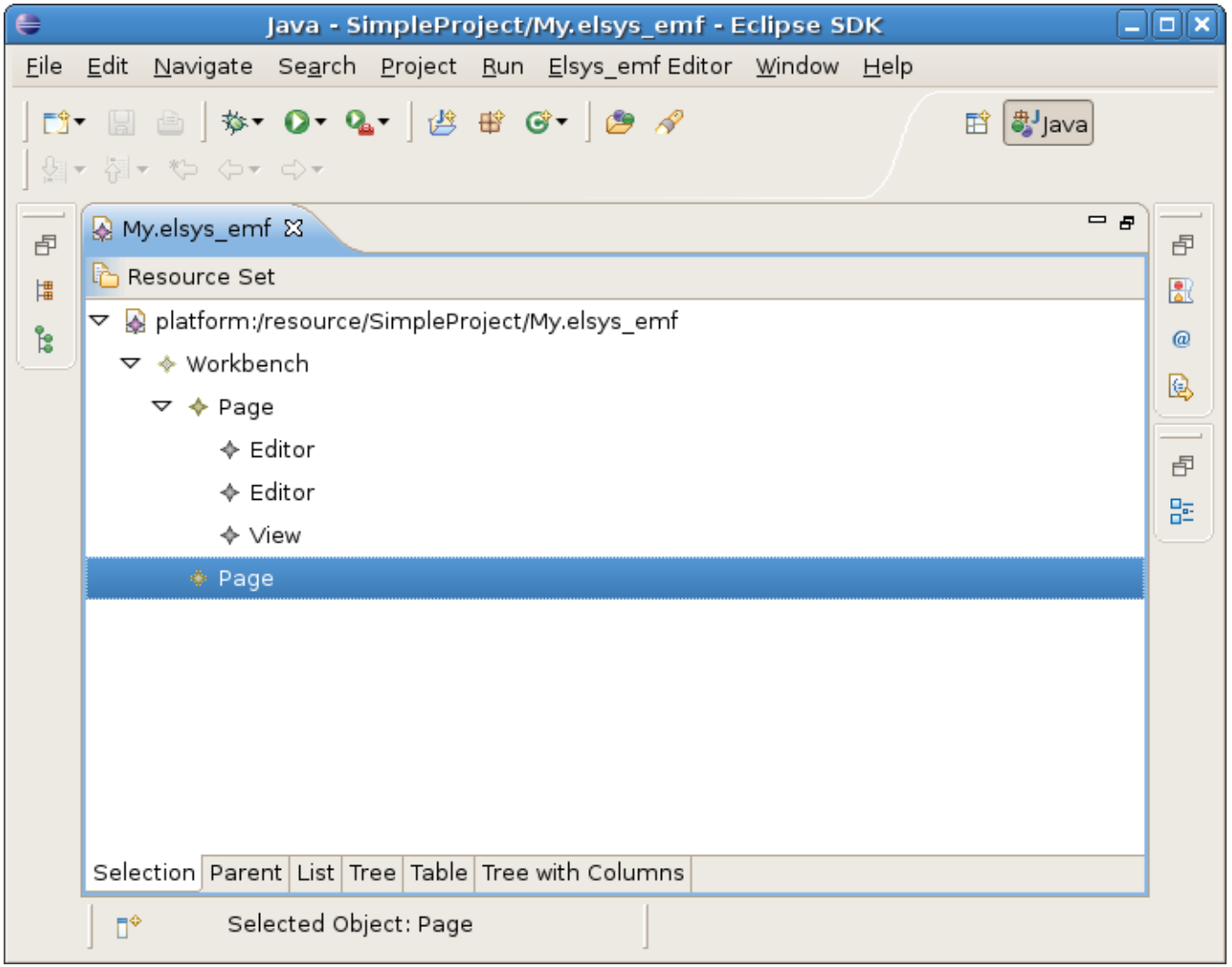
Eclipse Modeling Framework - Ecore

Създаденият ecore модел ще изглежда по следния начин:



Eclipse Modeling Framework - Редактор

Създаденият редактор ще изглежда по следния начин:



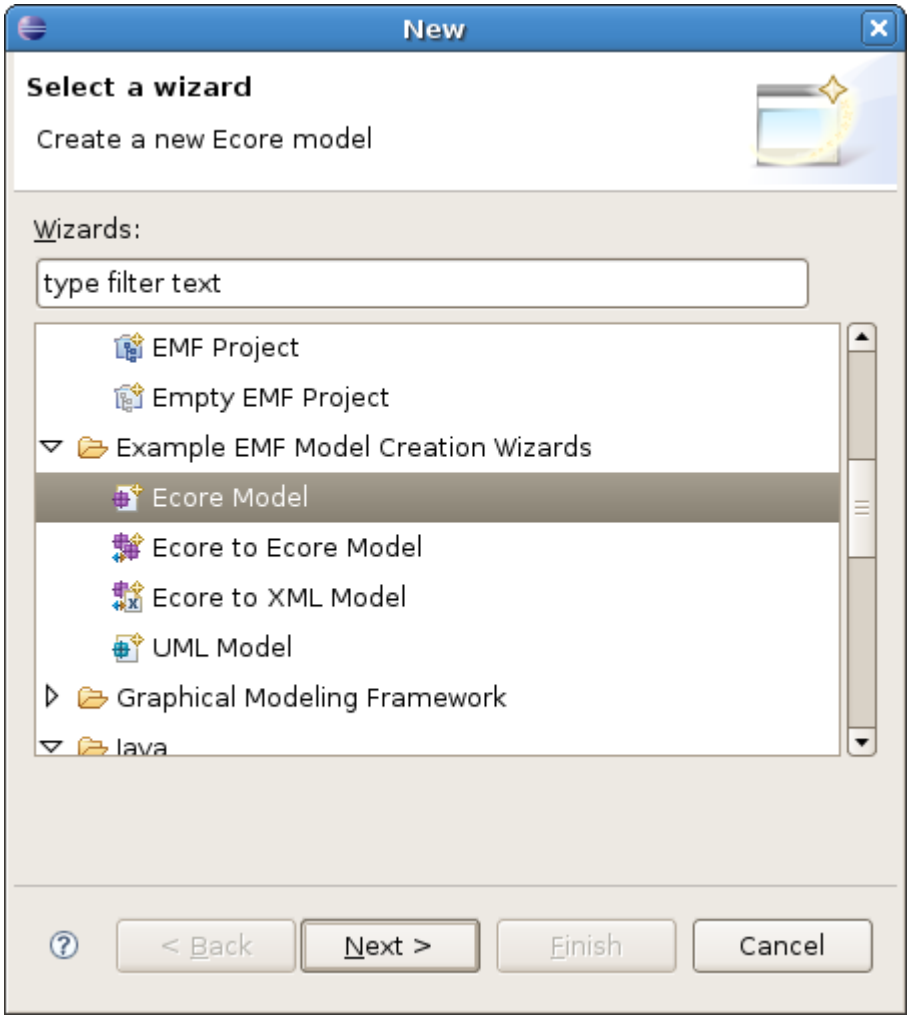
Ecore

Създаваме нов plugin проект
org.elsys.emf

Създаваме нова директоря
в проекта – model

Използваме магьосника за
създаване на ecore модели.
File -> New -> Other ->
Example EMF Model Creation Wizards
->Ecore

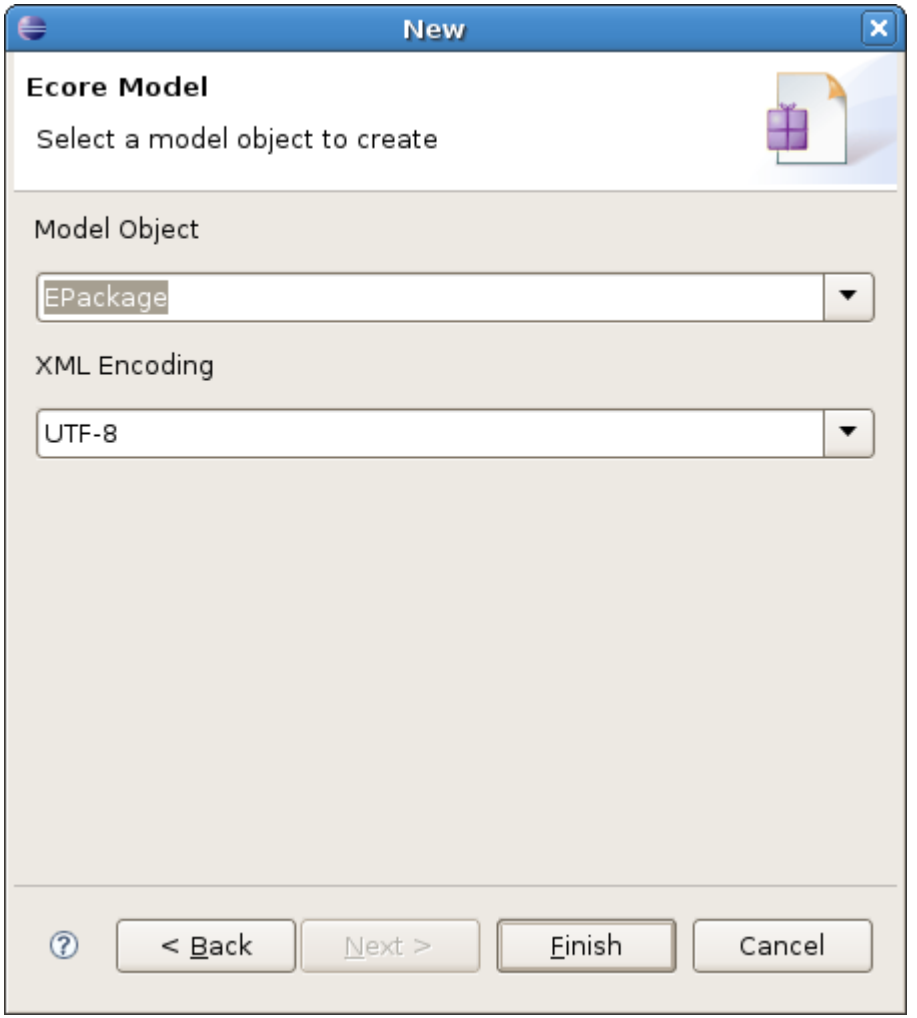
Посочваме име (emf_elsys.ecore)
и място (org.elsys.emf/model)
на създавания файл



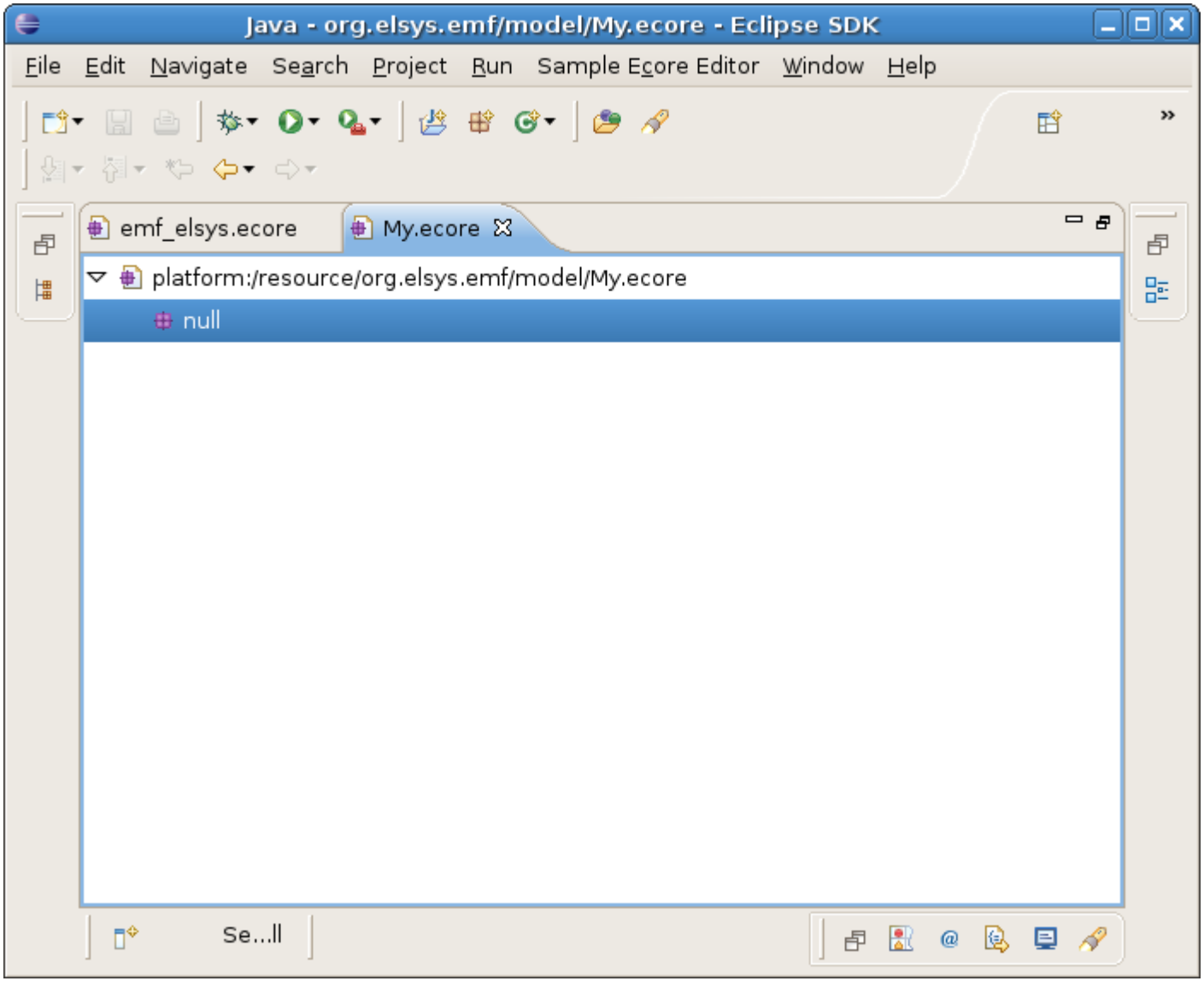
Ecore

Посочваме като Model Object обекта EPackage

Обектът от тип EPackage ще бъде родителят на всички елементи в модела. Те всичките ще се съдържат в него.

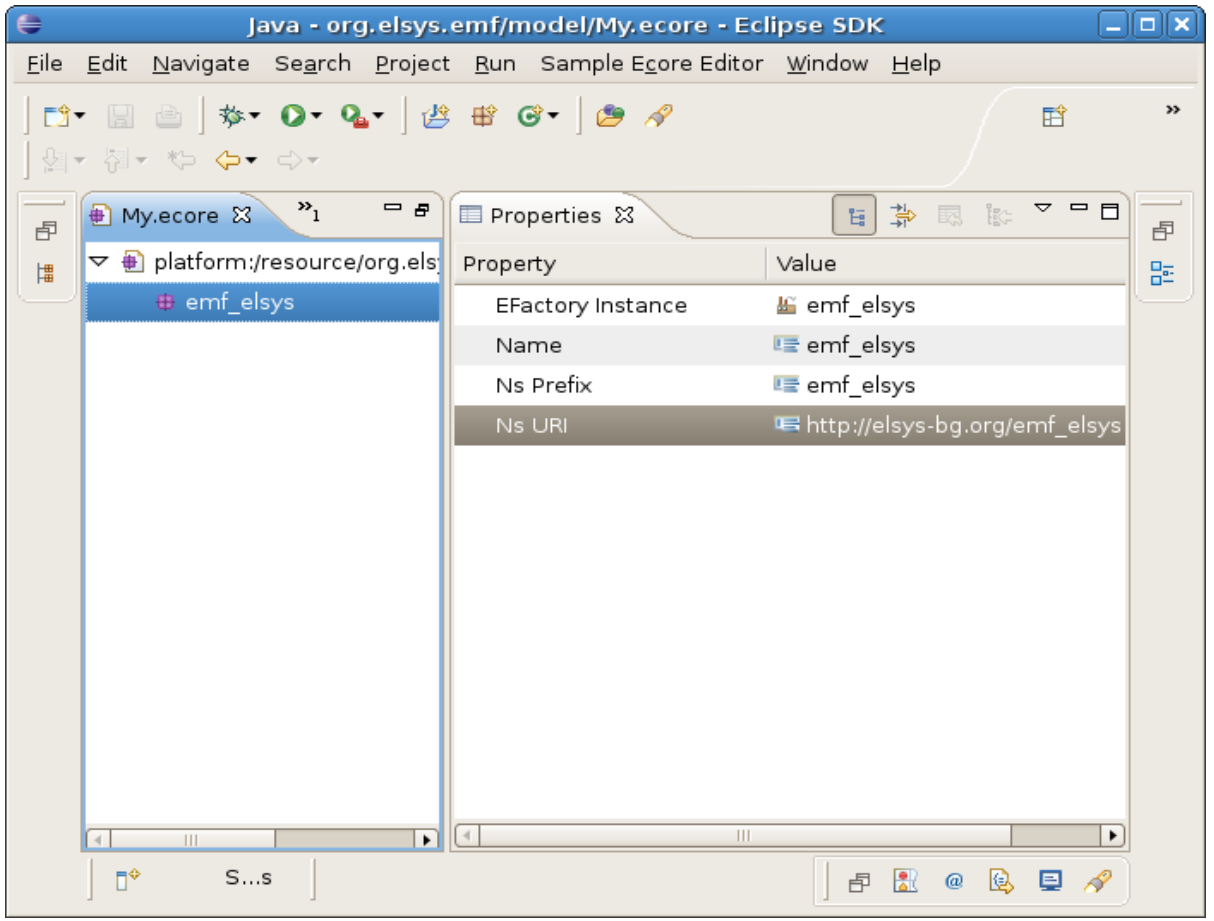


Създаденият ecore файл изглежда по следния начин:



Ecore

Активираме контексното меню върху обекта от тип Epackage. Избираме Show Properties View. Метамоделът има определени характеристики, които може да редактираме. Такава характеристика е името на пакета. Задаваме следните стойности: Name=emf_elsys, Ns Prefix=emf_elsys, Ns URI = http://elsys-bg.org/emf_elsys



Ecore

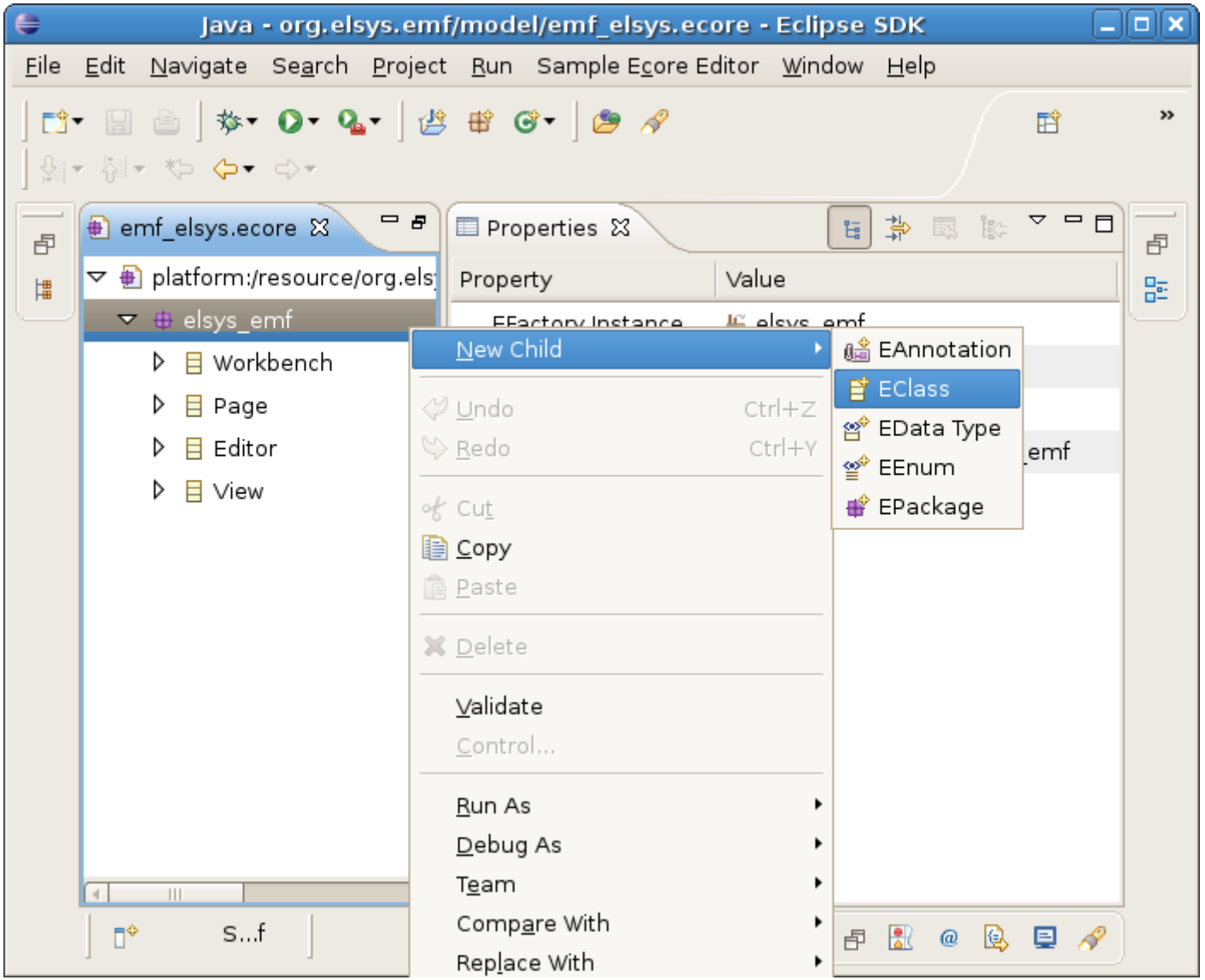
Предстои създаването на класове Workbench, Page, Editor, View. Тези обекти са от тип Eclass и ще бъдат деца на обекта от тип Epackage.

За да ги създадеме използваме контексното меню на обекта EPackage и избираме New Child -> EClass.

При създаването си класовете нямат име – името има стойност null. За да променим името отново използваме Properties изгледа.

Избираме обекта от тип EClass и променяме стойността на полето Name.

Ecore

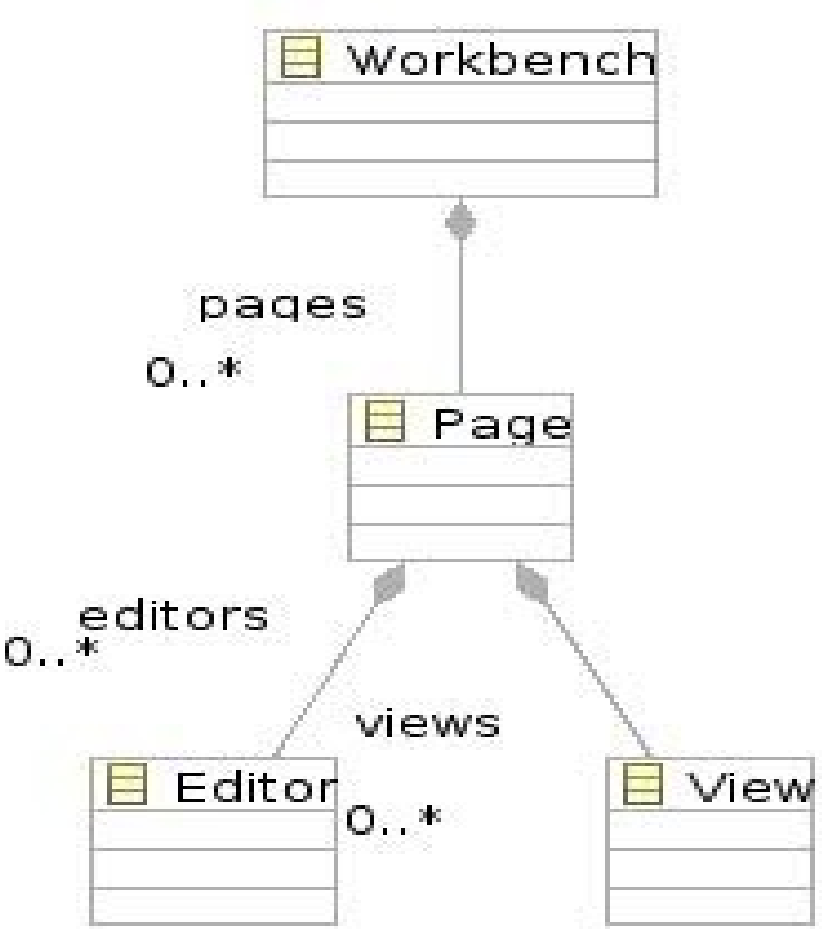

Java - org.elsys.emf/model/emf_elsys.ecore - Eclipse SDK
File Edit Navigate Search Project Run Sample Ecore Editor Window Help
emf_elsys.ecore
platform:/resource/org.elsy
elsys_emf
Workbench
Page
Editor
View
Properties
Property
Value
New Child
Undo Ctrl+Z
Redo Ctrl+Y
Cut
Copy
Paste
Delete
Validate
Control...
Run As
Debug As
Team
Compare With
Replace With
EAnnotation
EClass
EData Type
EEnum
EPackage
S...f

Ecore

Следвайки показаната схема класът Workbench трябва да съдържа в себе си списък от обекти от тип Page. За целта изпозваме контексното меню на класа Workbench и избираме New Child -> EReference

Използвайки Properties изгледа променяме атрибутите на референцията на следните

- Name: pages
- EType: Page
- Upper Bound: -1
- Containment: true



Ecore

- Name: pages – името на референцията. След като кодът бъде генериран така ще се казва списъкът в класа Workbench.
- EType: Page – посочва се какъв е типа на реферираните обекти.
- Upper Bound: -1 – Съществуват два атрибута Lower Bound и Upper Bound. Използвайки тези два атрибута може да се определят горна и долна граница на броя обекти, които може да има в списъка. EMF няма да забрани добавянето на допълнителни елементи. Но когато създаденият модел се валидира се проверява за правилната бройка елементи. В случая стойността „-1“ означава, че броят е неограничен.
- Containment: true – В даден момент моделът ще бъде запаметен и след това отново зареден от файл. Стойността на атрибута Containment показва, че списъкът pages „се съдържа“ в обекта Workbench и, че обектът Workbench трябва да се грижи за неговото създаване и унищожаване. pages ще бъде запаметен заедно със Workbench.

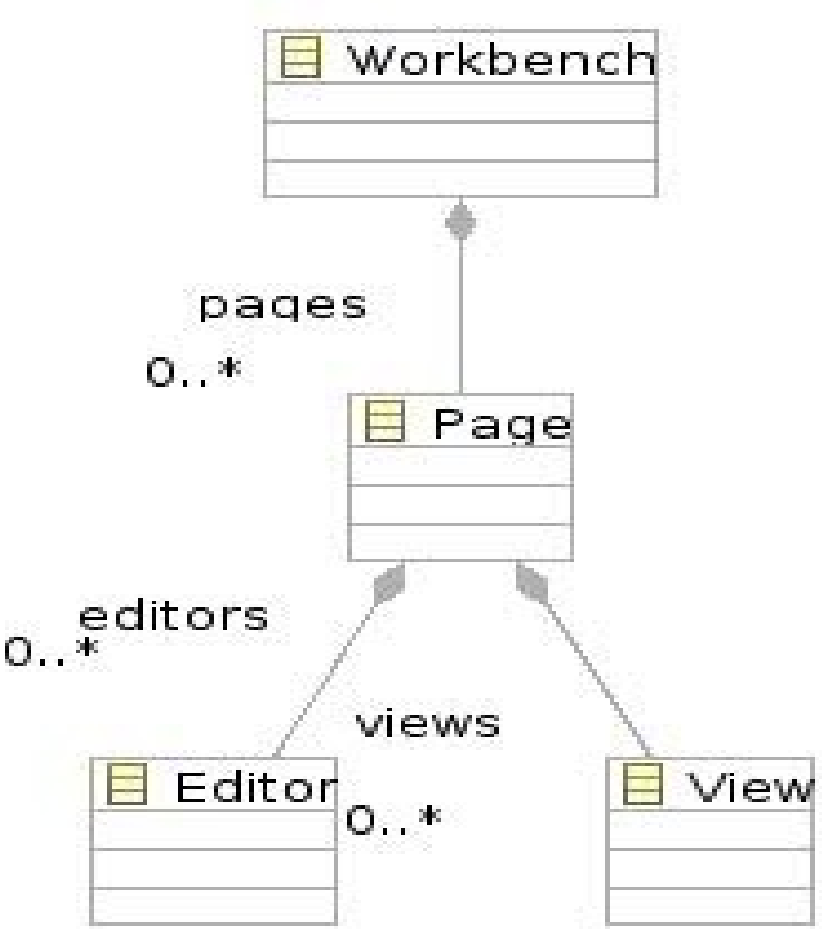
Ecore

Обектът Page ще знае точно в кой Workbench се намира. Това означава, че той трябва да има референция към него.

Използвайки контестното меню на Page избираме New Child->EReference.

Използвайки Properties изгледа променяме атрибутите на референцията на следните

- Name: workbench
- EType: Workbench
- EOpposite: pages: Page



Ecore

- Name: workbench – името на референцията
- EType: Workbench – типа на референцията
- EOpposite: pages: Page – Workbench съдържа списък с обекти от тип Page. Обектът Page знае кои е неговия Workbench. В случая Page може да има само един Workbench. Логично е добавянето на обект от тип Page към Workbench:pages автоматично да зададе стойност на Page:workbench. Обратното също е валидно. Ако не зададеме стойност на атрибута EOpposite то добавянето на Page към Workbench ще изглежда по следния начин:

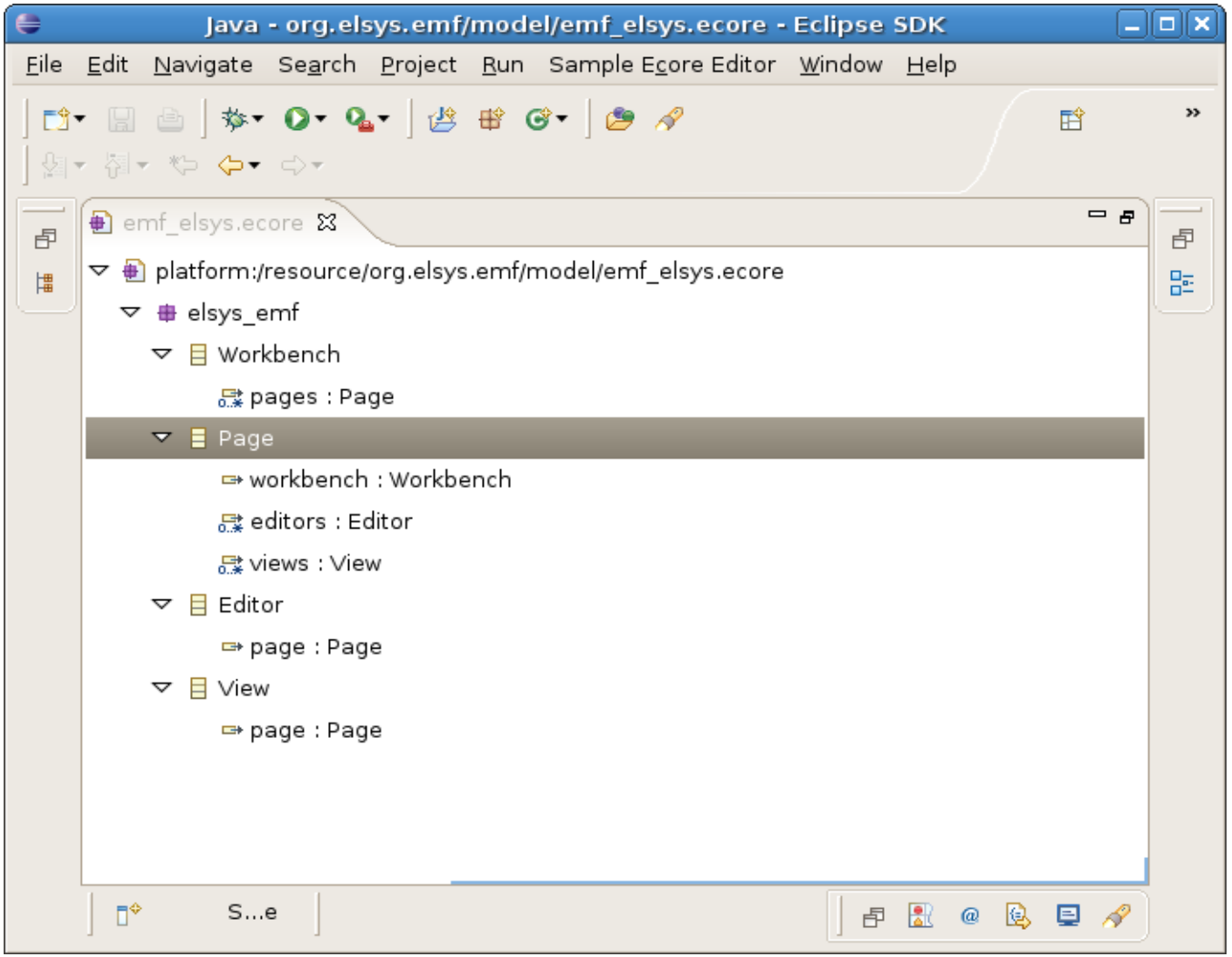
```
page.setWorkbench(workbench)
workbench.getPages().add(page)
```

В противен случай можем просто да извикаме

```
workbench.getPages().add(page).
```

Предимството е, че по този начин някой друг се грижи за управлението на сложни зависимости, което е от голяма полза при сложни модели.

Резултатът от дефинирането на всички обекти ще изглежда по следния начин:



Genmodel

Дефинирали сме ecore метамодела. Той представлява абстракция на бизнес логиката на нашето приложение. Тази абстракция е платформено и технологично независима.

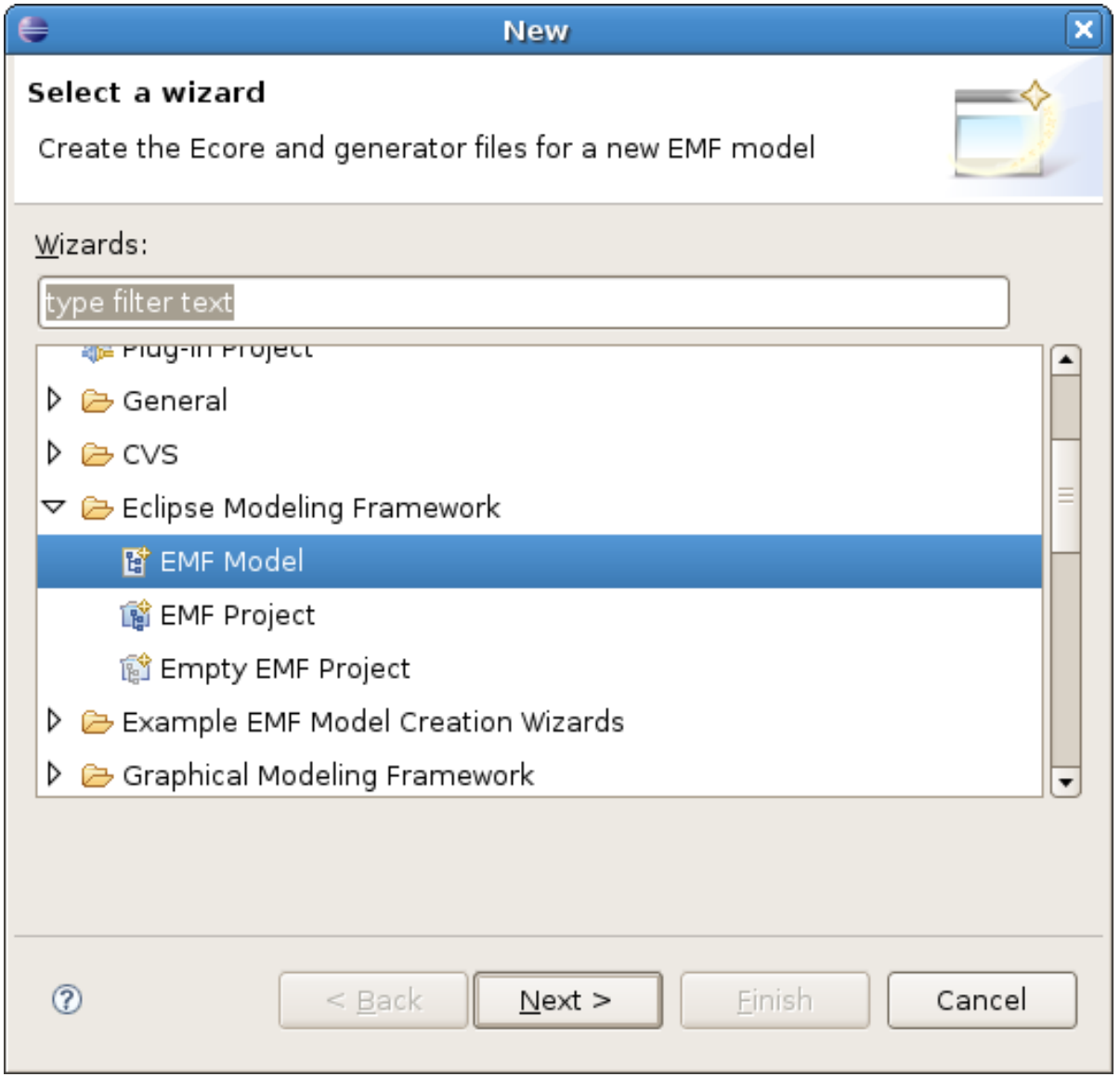
Искаме да създадем реално приложение работещо с описаната бизнес логика. Искаме това приложение да работи в Eclipse и да е написано на Java. Това означава, че трябва да конкретизираме нашия метамодел до модел разбиращ от Java и Eclipse. Използвайки този междинен метамодел ще генерираме реалната имплементация.

Наричаме този метамодел Genmodel

В него можем да зададеме характеристики като директорията, в която да се генерират класовете, пакетите в които да се намират, и др.

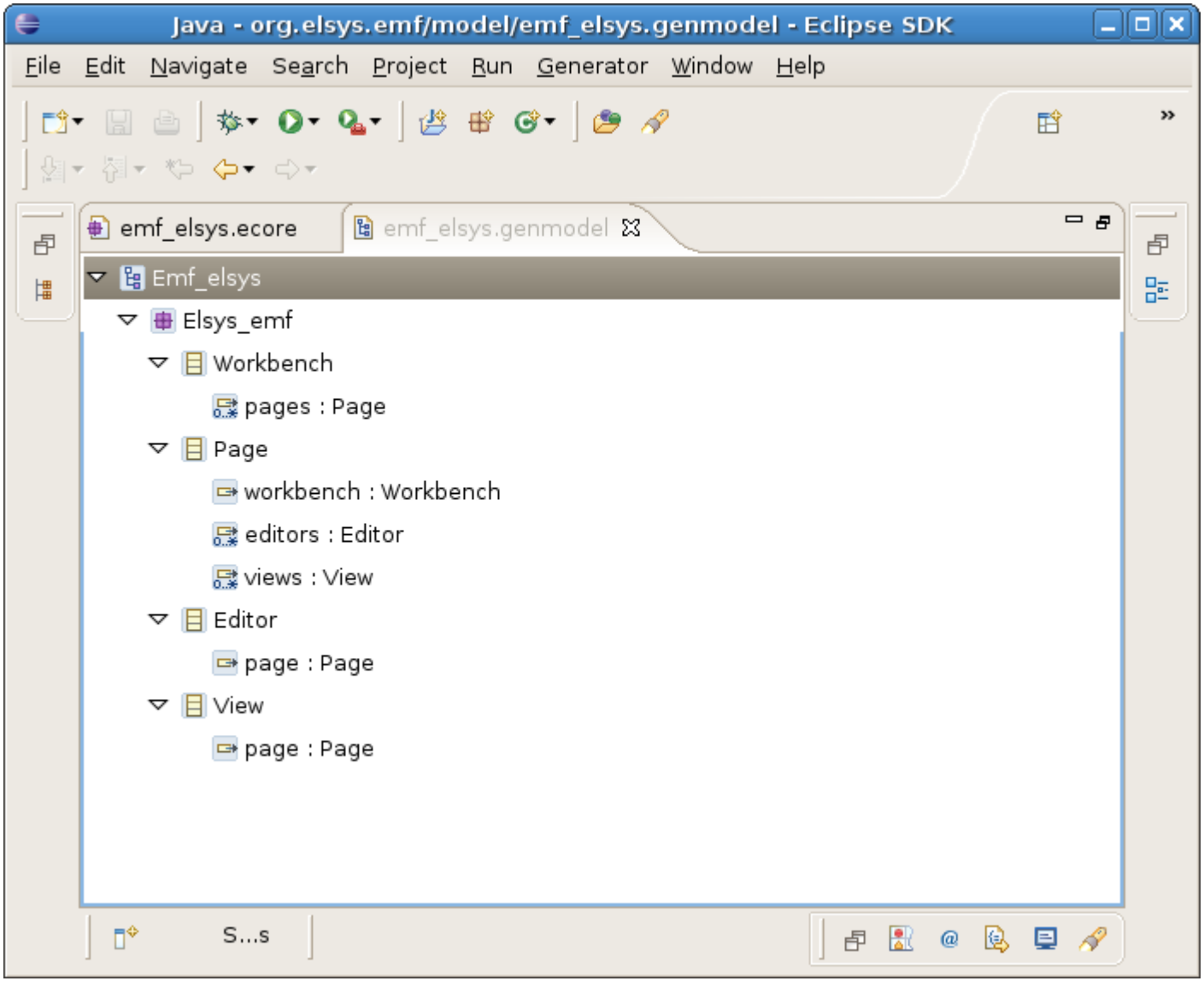
Създаването на genmodel става чрез File -> New -> Other -> Eclipse Modeling Framework -> EMF Model

Genmodel


New
Select a wizard
Create the Ecore and generator files for a new EMF model
Wizards:
type filter text
Plug-in Project
General
CVS
Eclipse Modeling Framework
EMF Model
EMF Project
Empty EMF Project
Example EMF Model Creation Wizards
Graphical Modeling Framework
< Back
Next >
Finish
Cancel

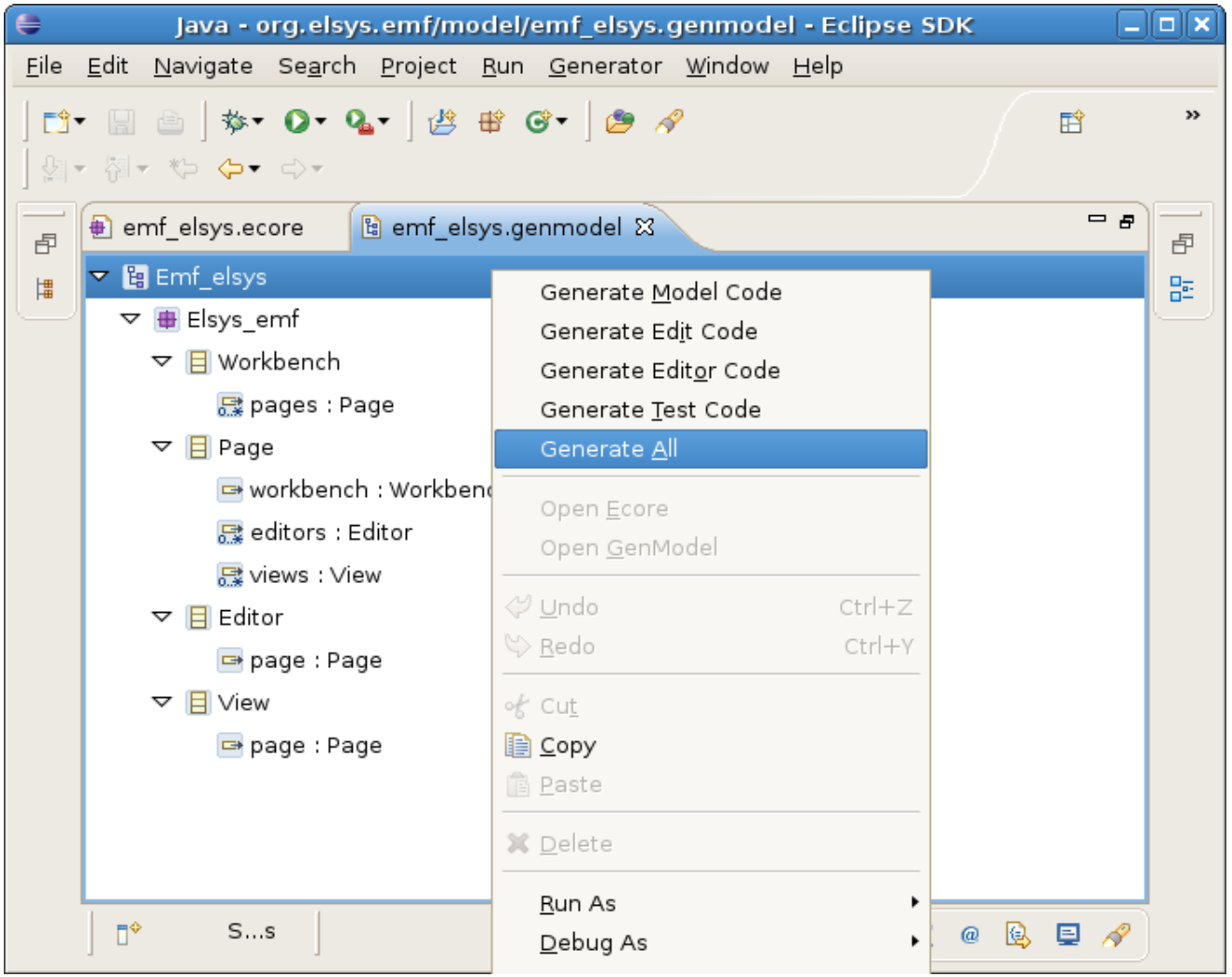
Genmodel

Създадения genmodel изглежда по следния начин:



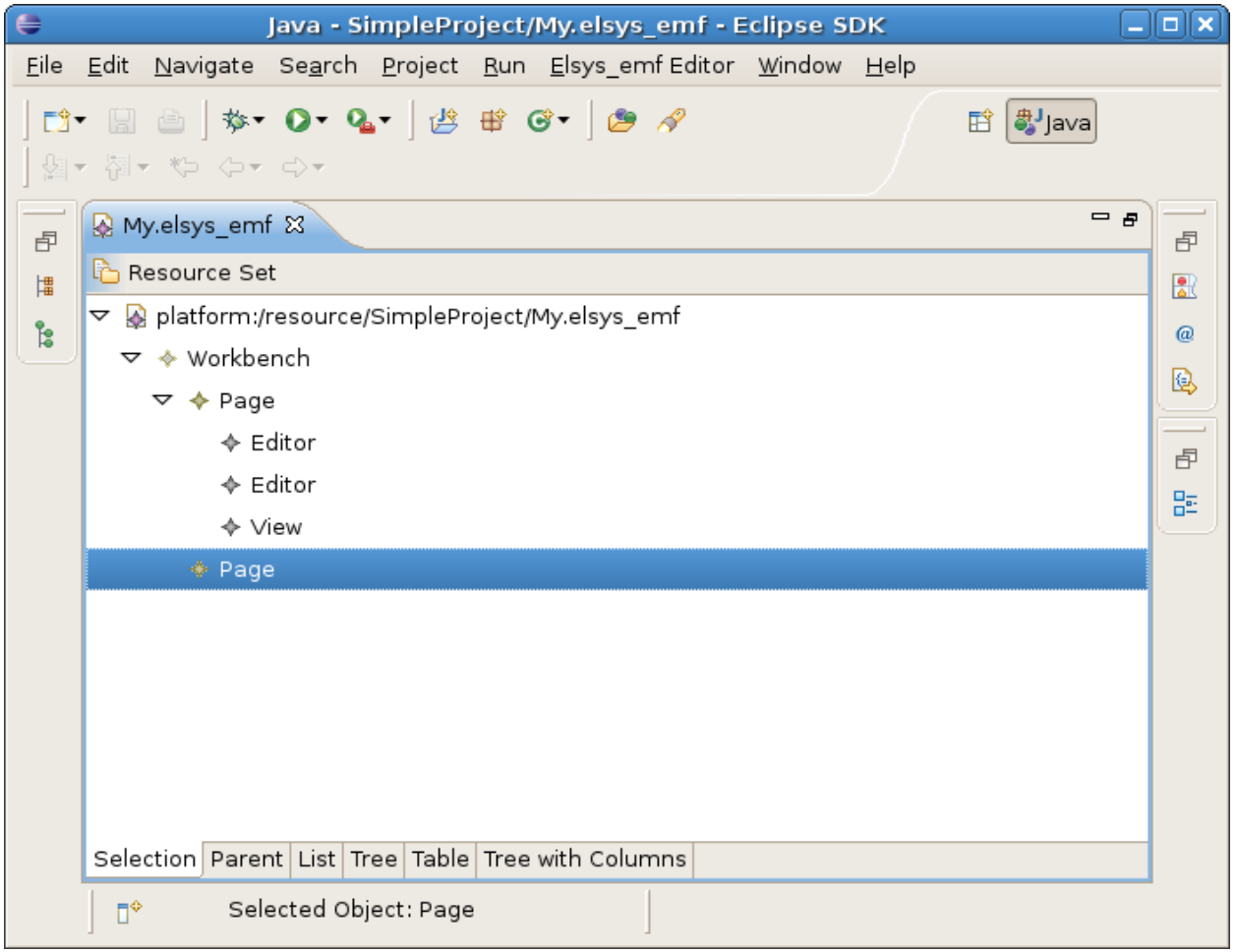
Genmodel

За да генерираме Java кода от контексното меню на главния обект в genmodel модела избираме Generate All. Това ще създаде още три plugin-а с имена *.edit, *.editor, *.tests



Editor

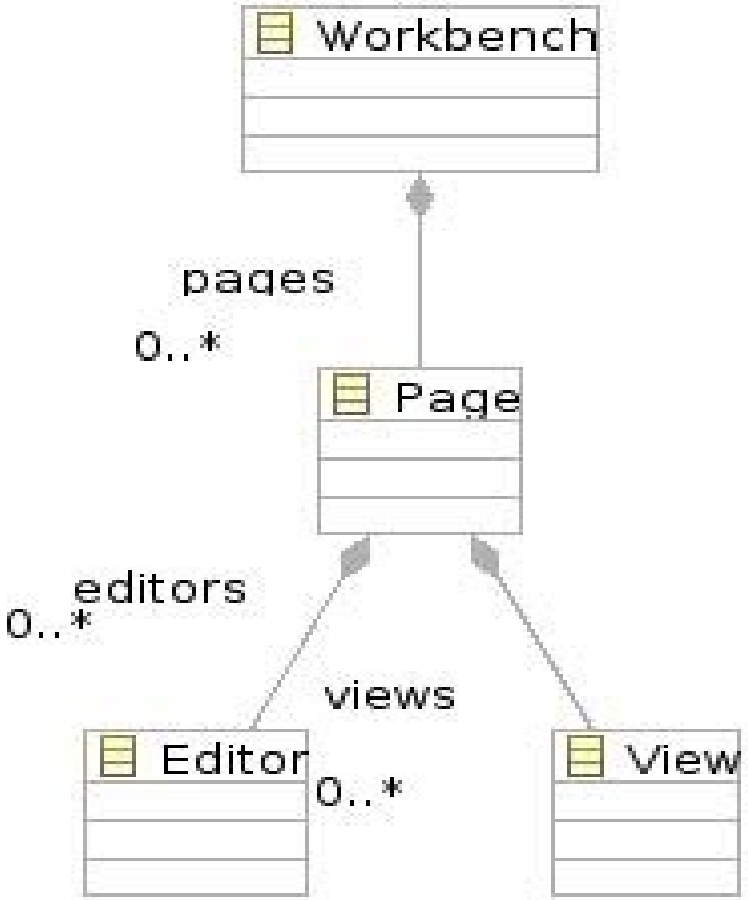
Генерираният редактор може да бъде използва за редактиране на файлове съдържащи описания модел. Стартираме Runtime Workbench и създаваме нов файл чрез File -> New -> Other -> Example EMF Creation Wizards - >Emf_elsys Model. Отвореният редактор изглежда по следния начин.



Editor

Упражнение:

Да се изгради Ecore моделът съгласно показаната диаграма. Да се генерира редакторът.



Copyright