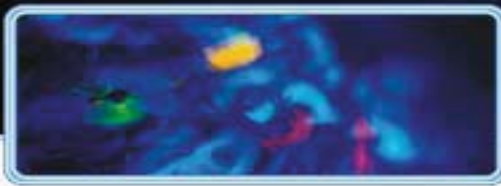


## A Little Terminology

- We use Eclipse-Terminology
- Workspace, projects, files, folders
  - Common place to organize & store development artifacts
- Workbench, editors, views, perspectives
  - Common user presentation and UI paradigm

# Menus, Views, Editors & Perspectives



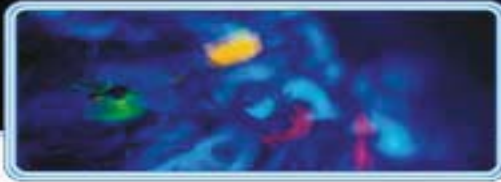
The screenshot shows the Eclipse IDE interface with the following components labeled:

- Menu bar**: Located at the top of the window, containing menus like File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help.
- Tool bar**: Located below the menu bar, containing icons for various actions like Save, Print, Undo, and Redo.
- Perspective and Fast View bar**: Located on the left side of the window, containing icons for switching between different IDE perspectives.
- Resource Navigator view**: Located on the left side, showing a tree view of the project structure (Example, bin, src, com, example, hw, HelloWorld).
- Properties view**: Located below the Resource Navigator, showing a table of properties for the selected file (HelloWorld.java).
- Message area**: Located at the bottom left, containing a search bar and a status bar.
- Text editor**: The central area showing the code for HelloWorld.java.
- Outline view**: Located on the right side, showing a tree view of the code structure (com.example.hw, HelloWorld, main(String[]) : void).
- Tasks view**: Located at the bottom center, showing a table of tasks (0 items).
- Bookmarks view**: Located at the bottom right, showing a list of bookmarks.
- Editor Status area**: Located at the bottom right, showing the current editor's status (Writable, Insert, 6 : 16).

Additional labels at the bottom of the screenshot:

- Stacked views**: Points to the bottom of the Resource Navigator and Properties views.
- Tasks view**: Points to the Tasks view.

# Perspective = Editors + Views



**Editor**

**Perspective**

**Views**

The screenshot shows the Eclipse IDE interface. A red dashed box highlights the entire workspace area, which is labeled as the **Perspective**. Within this perspective, the central area is labeled as the **Editor**, containing a Java code editor for `HelloWorld.java`. The code in the editor is:

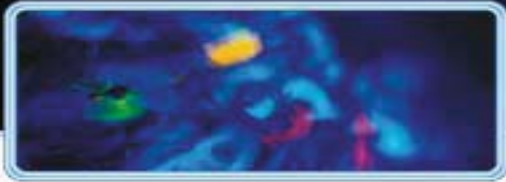
```
package com.example.hw;  
  
public class HelloWorld {  
  
    public static void main(String[] ar  
        System.out.println("Hello world")  
    }  
}
```

Surrounding the editor are several **Views**, which are also highlighted by a red dashed box. These include:

- Navigator**: A tree view showing the project structure with folders like `bin`, `src`, `com`, `example`, and `hw`, and files like `.classpath` and `.project`.
- Outline**: A view showing the class hierarchy, including `com.example.hw`, `HelloWorld`, and `main(String[]) : v`.
- Properties**: A table showing file properties for `HelloWorld.java`.

| Property      | Value            |
|---------------|------------------|
| editable      | true             |
| last modified | 8/28/02 9:10 PM  |
| name          | HelloWorld.java  |
| path          | /Example/src/com |
| size          | 144              |
- Tasks**: A table with columns for `C`, `!`, `Description`, `Resource`, and `In Folder`. It currently shows 0 items.
- Bookmarks**: A view for managing bookmarks.

The status bar at the bottom shows `Writable`, `Insert`, and `6 : 16`.



# NeOn Toolkit

Ontology Navigator

Entity Property View showing details

The screenshot displays the NeOn Toolkit interface with the following components:

- Ontology Navigator:** A tree view on the left showing the ontology structure. The 'Event' class is expanded, showing subclasses like 'Meeting' and 'ProjectMeeting'.
- Entity Property View:** The main window showing details for the 'Meeting' entity. It includes fields for Name, Namespace, and a table for Attributes and Relations.
- Instance View:** A small window at the bottom left, currently empty, used for viewing instances of the selected entity.

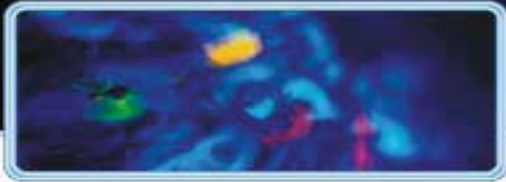
| Attribute | Range | Min | Max |
|-----------|-------|-----|-----|
|           |       |     |     |
|           |       |     |     |
|           |       |     |     |

| Relation     | Range  | Min | Max |
|--------------|--------|-----|-----|
| atEvent      | Event  | 0   | N   |
| hasPartEvent | Event  | 0   | N   |
| participant  | Person | 0   | N   |

| de | en | fr |
|----|----|----|
|    |    |    |

Instances





# NeOn Toolkit

Ontology Navigator

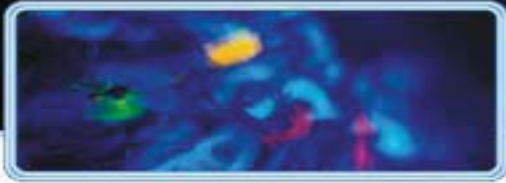
Other views e.g. visualizer

The screenshot displays the NeOn Toolkit interface with several key components:

- Ontology Navigator:** A tree view on the left showing the hierarchy of the ontology. The 'Document' class is selected under 'Concepts'. Other classes include 'Publication', 'Event', 'Organization', 'Person', 'Product', 'Project', 'Topic', 'Attributes', 'Relations', 'Rules', 'Mappings', and 'Queries'.
- Ontology Graph Visualizer:** The main central area showing a graph of the ontology. The 'Document' class is highlighted, with its 'attributes of Document' (identifier, coverage, title, date, description, type, format, subject, language, relation, source, rights) and its relationships (Unpublished, creator, publisher, contributor, organization, Publication) visualized. A legend at the bottom left identifies symbols for concept, ontology, attribute root, attribute, range concept, and relation.
- Instance View:** A small window at the bottom left, currently empty, used for viewing instances of the selected class.
- Navigation history:** A bar at the bottom right showing the current path: `"http://swrc.ontoware.org/"#ontology-07` → `Document`.

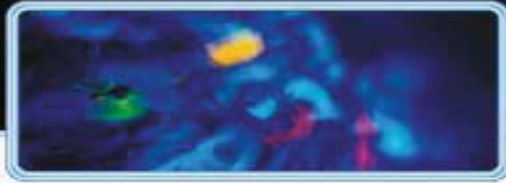
Instances





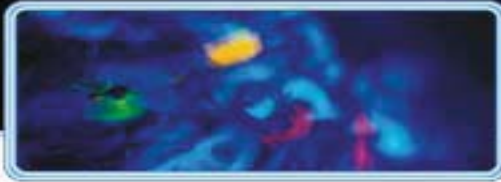
## Language Terminology

- **FLogic:**
  - Concepts
  - Attributes and relations
  - Instances
  
- **OWL:**
  - Classes
  - Data properties and object properties
  - Individuals
  
- **RDF(S):**
  - Classes
  - Properties
  - Instance



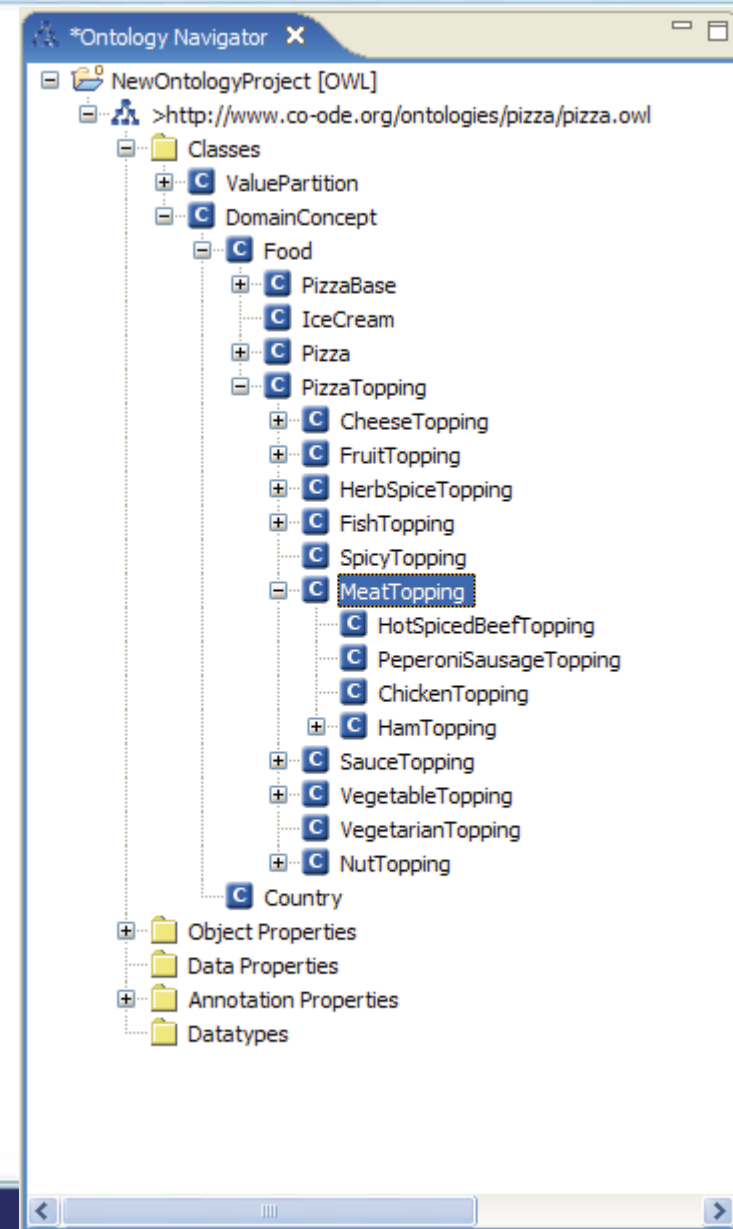
## Language Support of NeOn Toolkit

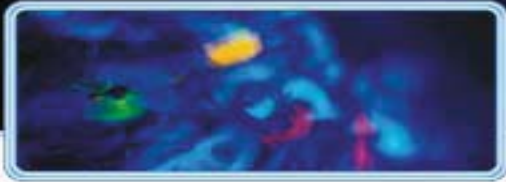
- The NeOn Toolkit follows the dual language approach.
- FLogic:
  - Native support for frame based language FLogic, which is declarative like Prolog, and object-oriented like Java;
  - Import and export of OWL and RDF(S) ontologies by translating to /from the frames-model.
- OWL:
  - Native support for managing OWL ontologies.
  - API is finished
  - GUI level support is work-in-progress



# Ontology Navigator

- **Workspace**
  - Hosts all ontology projects
- **Ontology-Project is the main structuring means**
  - Hosts ontologies in one language (OWL or FLogic)
  - Independent from other projects
  - Imported ontologies must be (are) in the same project
- **Ontology**
  - Hosts classes, properties, rules, ...
  - Depending on the ontology language
- **Folder metaphor**
  - Each folder contains different kinds of entities
  - Sometimes hierarchical structure





# Entity Property View

- Each type of entity has its specific entity property view

Entity Properties

Name: Mushroom

Namespace: <http://www.co-ode.org/ontologies/pizza/pizza.owl#>

Super Restrictions

| Quantifier | Property   | Range  | Min | Max |   |
|------------|------------|--|-----|-----|---|
| SOME       | hasTopping | TomatoTopping  |     |     | X |
| ALL        | hasTopping | [or MozzarellaTopping TomatoTopping MushroomTopping] |     |     | X |
| SOME       | hasTopping | MushroomTopping                                      |     |     | X |
| SOME       | hasTopping | MozzarellaTopping                                    |     |     | X |

Equivalent Restrictions

| Quantifier | Property | Range | Min | Max |  |
|------------|----------|-------|-----|-----|--|
|------------|----------|-------|-----|-----|--|

Super Classes

| Class Expression | Min | Max |   |
|------------------|-----|-----|---|
| NamedPizza       |     |     | X |

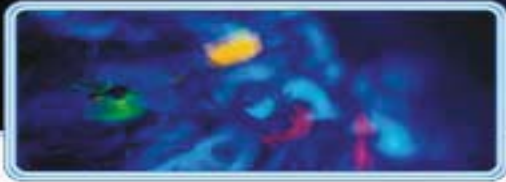
Equivalent Classes

| Class Expression | Min | Max |  |
|------------------|-----|-----|--|
|------------------|-----|-----|--|

Class Restrictions | Disjoint Classes | Annotations



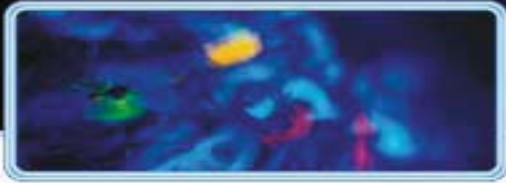




# Individuals

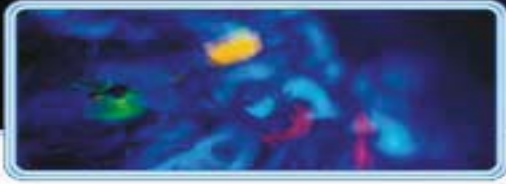
The screenshot displays the NeOn Toolkit interface with the following components:

- Ontology Navigator:** A tree view showing a project structure with several ontologies and a hierarchy of classes including ValuePartition, DomainConcept, Food, and Country.
- Entity Properties:** A panel for editing an individual, showing the namespace as 'Spain' and a specific URI. It includes a table for 'Descriptions' with columns for 'Property' and 'Value'.
- Individuals:** A list of individuals including Spain, England, Italy, France, America, and Germany.
- Same as / Different from:** Two panels for defining relationships between individuals, with 'Different from' currently listing England, Italy, France, Germany, and America.



# Agenda

- Introduction to the NeOn Toolkit
  - Terminology
  - Languages
  - Components
  
- **Modeling with the NeOn Toolkit**



# *Start of NeOn Toolkit*

## *Demonstration*