



# Nasdanika Codegen

Get Started

# Overview



- Concepts
- Installation & Update
- Create a modeling project
- Create a model & select Codegen View Point
- Populate the model
- Generate model
  - IDE
  - CLI
  - Server

# Concepts



- Generates one or more resources – files, directories
- Input data \* Generation Model => Generated resources
- Generation from
  - IDE
  - Command line
  - Server over HTTP
- Input data – Context
  - Properties – retrieved by name
  - Services – retrieved by type, e.g. ImportManager.class
  - Property & Service computers – compute data on retrieval
  - Can be constructed from:
    - YAML
    - JSON
    - *Properties files*
    - *Command line options*
    - *External sources – context builders*
- Interpolation
  - Replacement of \${<key>} token with value from the context
  - Widely used in generator models
- Iteration – execution of generation zero or more times based on the type and value of the iterator context property

```
Usage: nsd codegen [-fhjV] [-b=<baseUri>] [-o=<outputDir>]
                   [-p=<progressOutput>] [-P=<parallelism>] [-t=<timeout>]
                   [-B=<URL>]... [-c=<String=String>]... [-C=<URL>]...
                   [-M=<String=String>]... [-R=<URL>]... URI
Generates resources from a codegen model
URI          Model (object) URI resolved relative to the
             current directory. May include fragment to
             address a non-root object
-b, --base-uri=<baseUri> Base URI for resolving and relativizing other
                           URI's. Resolved against the output directory
                           URI. Defaults to the output directory URI.
-B, --context-builders=URL Context builders configuration resource URL
                           relative to the current directory. See online
                           documentation at https://www.nasdanika.
                           org/builds/develop/doc/reference/cli/context-buil
                           ders.html for details.
-c, --context-entry=<string=String> Context entries.
                                         Shadow entries in contexts and mounts.
-c, --context=URL Context resource URL relative to the current
                     directory. YAML, JSON, or properties. In
                     properties dots are treated as key path
                     separators. Type is inferred from the content
                     type header, if it is present, or extension.
                     Contexts are composed in the order of
                     definition, later context entries shadowing the
                     former
-f, --file      URI is a file path
-h, --help       Show this help message and exit.
-j, --json       Output progress in JSON
-M, --context-mount=<String=String> MappingContext resource URL relative to the
                                         current directory. YAML, JSON, or properties. In
                                         properties dots are treated as key path
                                         separators. Type is inferred from the content
                                         type header, if it is present, or extension.
                                         Mounts shadow context entries.
-o, --output=<outputDir> Output directory, defaults to the current directory
-p, --progress=<progressOutput> Output file for progress monitor
-P, --parallelism=<parallelism> If the value greater than one then an executor
                                         service is created and injected into the context
                                         to allow concurrent execution.
-r, --resource=URL URL of a resource to load to the resource set
                           relative to the current directory.
-t, --timeout=<timeout> If parallelism is greater than one this option
                           specifies timeout in seconds awaiting completion
                           of execution. Default value is 60.
-v, --version    Print version information and exit.
Exit codes:
0 Success
1 Unhandled exception during execution
2 Invalid input
3 Diagnostic failed
4 Execution failed or was cancelled, successful rollback
5 Execution failed or was cancelled, rollback was not successful
6 Executor service termination timed out
```

# Generation Model



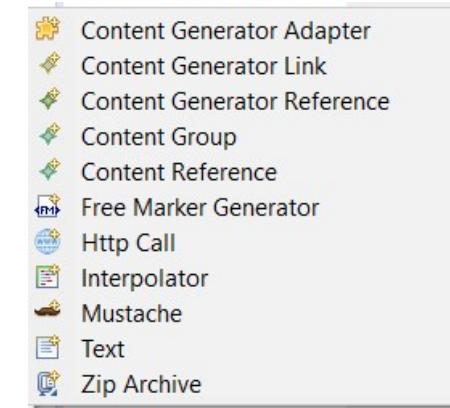
- A tree of resource generators and their contents generators
- Multiple representations per model
- Models can reference other models
- Resource generators:
  - Container – directory/folder
  - File
  - Adapter – calls Java extension to generate resources
  - Link – loads resource generator from URL,
    - *Reuse*
    - *Polymorphic generation with interpolation – link different models based on input*
  - Reference – points to a generator, e.g. a generator from another model
    - *Reuse*
  - Group – a construct for configuration and iteration
  - Zip resource collection – loads resources from a zip stream, which can be loaded by, say, HTTP Call
    - *Leveraging external generators*
- Reconciliation with pre-existing resources
  - Several options, default is Overwrite
  - Optional support for merging
    - *Ability to change generator input, re-generate and preserve manual modifications*
    - *Concurrent evolution of a generator and code generated by it – improve both at the same time and then combine (merge).*

The screenshot shows the Eclipse IDE interface with the 'eclipse-maven-project.codegen' project open. The left pane displays a tree view of the project structure, including 'Eclipse Maven Project', 'Configuration', 'Elements', 'src' (containing 'Configuration', 'Elements', and 'main' folder), and 'Content' (containing '\$(java/root-package)', 'Elements', 'Content', 'Body', 'Member Group Properties', 'Configuration', 'Elements', and 'Constructor'). The right pane shows the 'Properties' view for the '\$(java/root-package)' element. The 'Main' tab is selected, showing fields for 'Title', 'Id' (set to '7456798f-7e4b-4076-96e7-7f58aa62fe14'), and 'Enabled' (checkbox checked). The 'Properties' tab is also visible. Below the properties view, there are tabs for 'Reconcile Action' (radio buttons for Keep, Append, Merge, Overwrite, Cancel) and 'Merger' (text input field for 'Action to take if resource with given name already exists'). A legend on the right side defines icons for various resource types: Container (green folder), File (blue document), Resource Generator Adapter (yellow gear), Resource Generator Link (pink star), Resource Generator Reference (orange star), Resource Group (blue folder with gear), Source Folder (brown folder), and Zip Resource Collection (yellow folder with gear).

# Content Generators



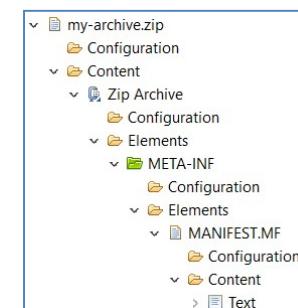
- Adapter – calls Java extension to generate content
- Link – loads content generator from URL
  - Reuse
  - Polymorphic generation with interpolation
- Generator Reference – points to a generator, e.g. a generator from another model
  - Reuse
- Group – a construct for configuration and iteration
- Content reference – loads content from a URL
- Free Marker Generator – uses Apache Free Marker
- HTTP Call
  - HTTP methods
  - Headers
  - Body
  - Response as content, can be interpolated
- Interpolator – expands \${<key>} tokens.
- Mustache – evaluates Mustache template
- Text – static text with support of interpolation
- Zip archive – constructs Zip stream from contained resources



Properties

https://nasdanika.org/builds/develop/index.html

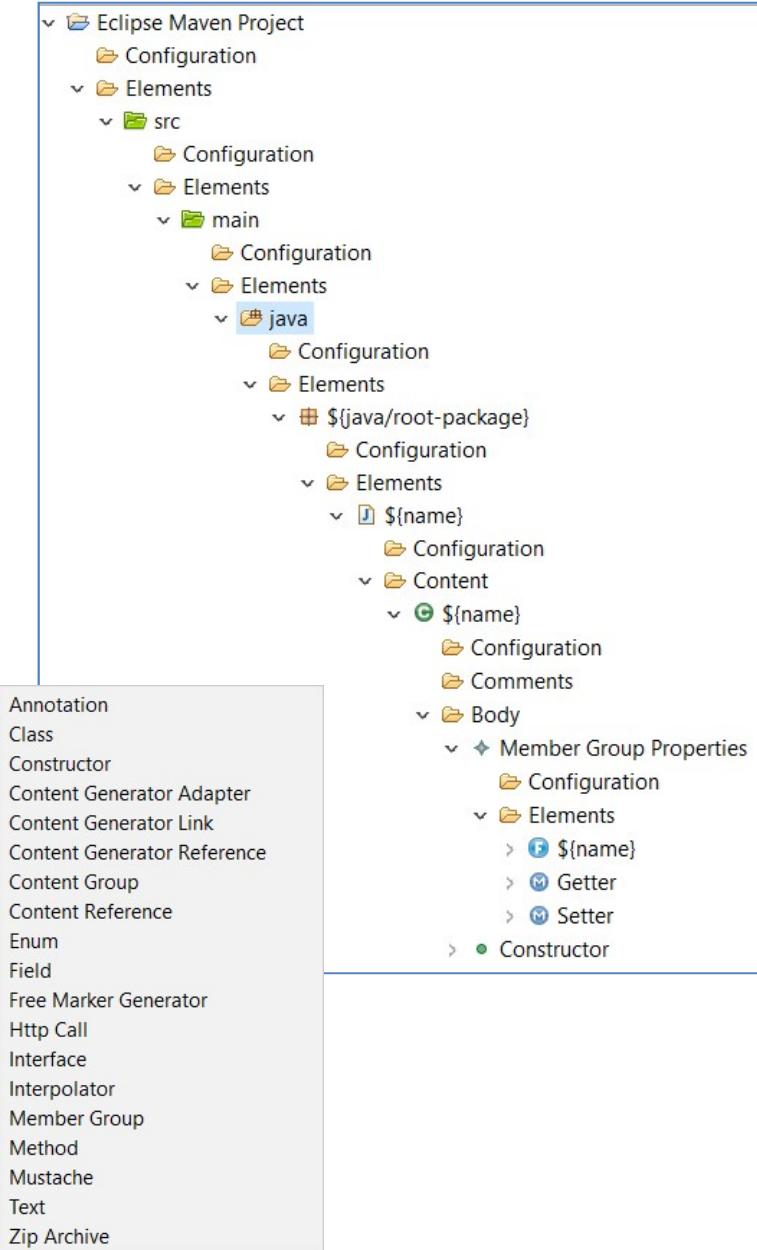
Main	Properties
Description	
Code	
Title:	<input type="text"/>
Id:	<input type="text"/> 295ae0d6-49a9-4fb9-b7bb-9ce19de5e49b
Enabled:	<input checked="" type="checkbox"/>
Iterator:	<input type="text"/>
Url:	<input type="text"/> https://nasdanika.org/builds/develop/index.html
Method:	<input checked="" type="radio"/> GET <input type="radio"/> POST <input type="radio"/> PUT <input type="radio"/> DELETE <input type="radio"/> PATCH
Connect Timeout:	<input type="text"/> 60
Read Timeout:	<input type="text"/> 60
Success Code:	<input type="text"/> 200
Interpolate:	<input type="checkbox"/>



# Java Support



- Resource generators:
  - Source folder – a specialized container which can contain packages
  - Package – a specialized container
    - *Can contain Compilation Units*
    - *Uses Java naming for packages – dot-separated*
  - Compilation Unit – a specialized file
    - *Generates package declaration*
    - *Import manager*
      - Use \${import/<fully qualified type name>} to replace FQN with short name
      - Generates imports after the package declaration
    - *Can contain types – Annotations, Classes, Enums, Interfaces*
    - *Source formatting*
    - *Native merger (JMerger from the EMF code generator)*
      - Regenerates only members with a “clean” @generated Javadoc tag – no value
      - Keeps members without the @generated tag or with a “dirty” tag, e.g. @generated NOT
- Content generators:
  - Types – specialized content generators. Some can contain members.
  - Members – types, fields, methods, and constructors

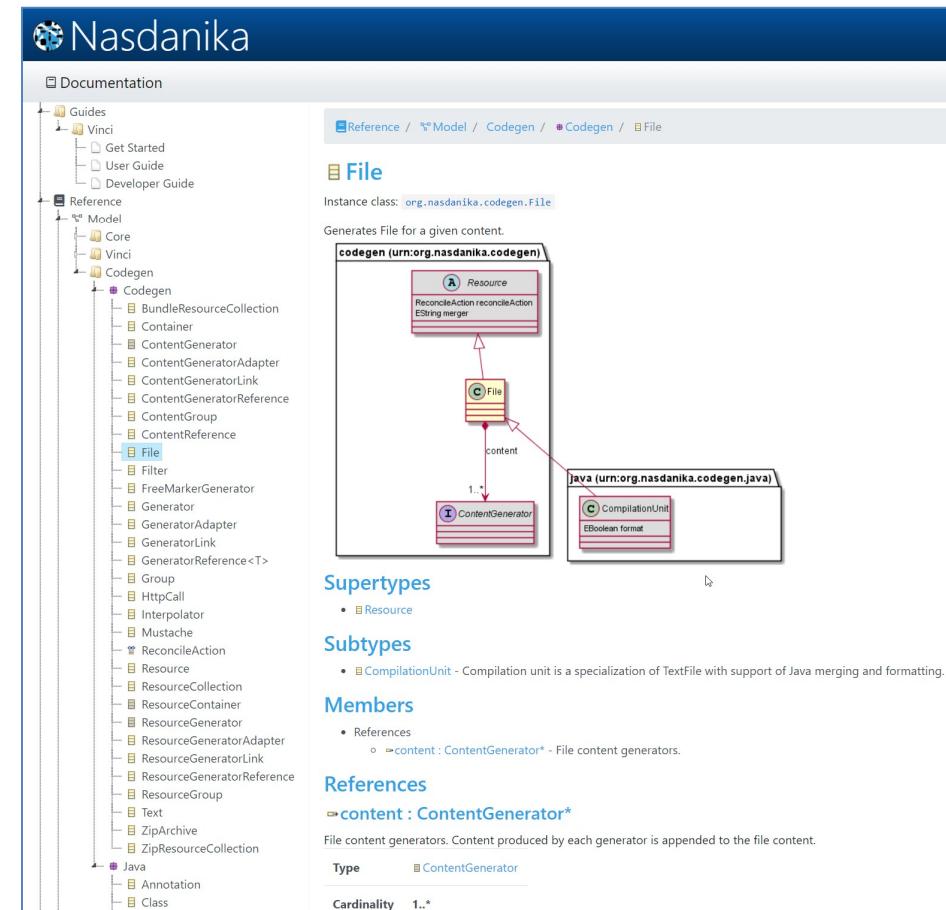


# Documentation



- Online
- Integrated Eclipse Help
- Model reference
  - The primary source of information on how to use model elements
  - First sentence of attribute documentation is shown as a tooltip in properties view
- Command line reference

The screenshot shows the Eclipse Help interface for the 'codegen' command. The left sidebar contains navigation links for Guides, Reference, and Command Line Interface. Under 'Command Line Interface', there is a section for 'nsd' which includes 'codegen'. The main content area displays the usage of the 'codegen' command, its options, and detailed descriptions for each option. The 'Usage' section shows the command line syntax: `nsd codegen [-h|y] [-b=baseUrl] [-o=outputDir] [-p=progressOutput] [-s=parallelism] [-t=timeout]`. Below this, the 'Generates resources from a codegen model' section provides a detailed explanation of the 'URI' parameter, which is described as a resolved URL relative to the current directory or a non-root object. Other parameters like '-b', '-o', '-p', '-s', '-t', and '-y' are also explained.



# Installation prerequisites



- Microsoft Windows 10 or 7
- Permissions
  - For binary packages:
    - *Download zip files*
    - *Extract zip files*
    - *Launch executables*
  - For installing into an existing Eclipse:
    - *Eclipse installation, Eclipse Modeling package is recommended.*
    - *Access to internet from Eclipse to install from the Eclipse Marketplace or Install New Software*
    - *Or download zip files (p2 archive)*

# Installation



- Downloadable packages:
  - Nasdanika Tools Suite for Windows – based on Eclipse modeling 2020-03
  - Nasdanika Tools Suite for Windows & Open JDK 8 bundle
- Installing into an existing Eclipse:
  - Eclipse Marketplace
  - P2 repository (update site),
  - Archived P2 repository
    - *Download and install from archive*
    - *Team environment - extract and host on the intranet*

**Nasdanika**

Nasdanika specializes in creating tools and solutions for Model-Driven development and code generation.

**Products**

**Vinci**  
Nasdanika Vinci is a visual modeling tool for creating models of web sites and then generating static web sites from the models.

- Get Started
- User Guide
- Developer Guide

**Ресурсы на русском (Resources in Russian)**

- Создайте веб-портфолио с генератором сайтов Nasdanika Vinci - три варианта:
  - Юдеми курс (Udemy course).
  - Веб сайт (Web site) - создан в Винчи.
  - Ютуб лист (YouTube list).

**Packages**  
We provide our products in a form of a Eclipse package - Nasdanika Tool Suite. We also provide another package - Nasdanika Developer Tools - a collection of third-party tools extending the Eclipse Modeling package.  
By downloading Nasdanika packages you are agreeing to the [Terms of Use](#).

**Tool Suite**

- Packages:
  - Nasdanika Tool Suite for Windows JDK bundle - a self-contained package which includes Zulu Community OpenJDK 8
  - Nasdanika Tool Suite for Windows - requires Java Runtime Environment
- **Install** - drag into your running Eclipse workspace. Requires Eclipse Marketplace Client. It is recommended to install into the Eclipse Modeling Tools package for the best modeling experience in the Modeling Perspective.
- P2 repository (update site)
  - URL: <https://nasdanika.org/builds/develop/packages/tool-suite/repository>
  - Archived

**Eclipse Marketplace**  
Select solutions to install. Press Install Now to proceed with installation.  
Press the "more info" link to learn more about a solution.

Search Recent Popular Favorites Installed Giving IoT an Edge  
Find: Nasdanika tool suite All Markets All Categories Go

**Nasdanika Tool Suite 2020-06**  
Nasdanika Tool Suite provides tooling for model-driven development (MDD) and code generation. Currently the Tools Suite features Nasdanika Vinci - an Eclipse... [more info](#)  
by Nasdanika LLC, EPL 2.0

Installs: 0 (last month) **Install**

**Spring Tools 4 (aka Spring Tool Suite 4) 4.7.1.RELEASE**  
Spring Tools 4 is the next generation of Spring Boot tooling for your favorite coding environment. Largely rebuilt from scratch, it provides world-class support... [more info](#)  
by VMware, EPL

Installs: 1.62M (32.494 last month) **Install**

**Ethernet POWERLINK XDD Tool Suite 1.1.1**  
The plugin include functionality to check CN XDD / XDC files against the DS311 V1.2.0 and the DS 301 1.3.0 specification. Another plugin enables the user to... [more info](#)  
by B&amp;R Industrial Automation GmbH, BSD

**Marketplaces**

?

< Back | Install Now | Finish | Cancel

**Available Software**  
Check the items that you wish to install.

Work with: <https://nasdanika.org/builds/develop/packages/tool-suite/repository> Add... Manage...

Type filter text:

Name	Version
✓ Nasdanika	2020.9.0.202008120718
✓ Tools	
✓ Nasdanika Tool Suite	2020.9.0.202008120718

1 item selected

**Details**  
Nasdanika and third-party tools.

Show only the latest versions of available software  Hide items that are already installed  
 Group items by category  What is [already installed?](#)  
 Show only software applicable to target environment  
 Contact all update sites during install to find required software

?

< Back | Next | Finish | Cancel

# Updating



- First time
  - Help > Install new software
  - Work with – P2 site URL
- On-going – Help > Check for Updates

**Available Software**  
Check the items that you wish to install.

Work with:  Add... Manage...

Type filter text

Name	Version
Nasdanika	2020.9.0.202008120718
Tools	
Nasdanika Tool Suite	

Select All Deselect All

1 item selected

Details

Show only the latest versions of available software    Hide items that are already installed  
[What is already installed?](#)

Group items by category    Show only software applicable to target environment  
 Contact all update sites during install to find required software

?

< Back Next > Finish Cancel

# Create a modeling project and a model



- File > New > Modeling Project
- File > New > Other
- Nasdanika > Codegen Model
- Select Model Object – Resource Group

The first screenshot shows the 'Select a wizard' step of the 'New' wizard, with 'Codegen Model' selected under the 'Nasdanika' category. The second screenshot shows the 'Codegen Model' step, where 'Resource Group' is selected as the model object to create.

- Select Codegen Viewpoint

The screenshot shows the 'Viewpoints Selection' dialog with 'Selected viewpoints' listed. Under 'Selected viewpoints', there is a checkbox labeled 'Codegen'. At the bottom of the dialog are 'OK' and 'Cancel' buttons.

The screenshot shows the Eclipse IDE interface with the 'File' menu open. The 'Modeling Project' option is highlighted. The 'Model Explorer' view is visible at the bottom left, showing a project named 'codegen-demo' with 'Viewpoints Selection' and 'Create Representation' items. A context menu is open over the 'codegen-demo' project, with 'New' selected. A submenu for 'New' is open, showing options like 'Project...', 'File', 'Folder', 'Modeling Project', 'Example...', and 'Other...'. The 'Other...' option is highlighted.

# Modeling – create representations



- By default the generated editor is used
- It is recommended to use representations
  - Expand the model in the Model Explorer
  - Right-click on a model element of interest
  - New Representation > new Codegen Adapter Factory Tree
  - Name the representation
  - One model can have multiple representations
  - Opens a representation editor once created
  - Going forward double-click on a representation to open it

The screenshot shows the Eclipse Modeling Environment. On the left, the Model Explorer view displays a project structure with 'codegen-demo' expanded, showing 'Project Dependencies', 'demo.codegen' (with 'Resource Group' selected), 'representations' (with 'New Representation' selected), and 'RemoteSystemsTe'. In the center, the 'demo.codegen' view shows a 'Resource Set' with a 'platform:/resource/codegen-demo/demo.codegen' entry containing a 'Resource Group'. A context menu is open over the 'New Representation' item, with 'new Codegen Adapter Factory Tree' highlighted. To the right, a 'New Codegen Adapter Factory Tree' dialog box is open, showing a 'Name:' field with 'Demo' typed in. At the bottom right of the dialog are 'OK' and 'Cancel' buttons. Below the dialog, another Model Explorer view shows the 'Resource Group' node under 'demo.codegen' now contains a sub-node named 'Demo'.

Important – do not edit properties of elements selected in the Model Explorer. These edits might be lost. Select elements in a representation editor.

# Modeling – populate the model



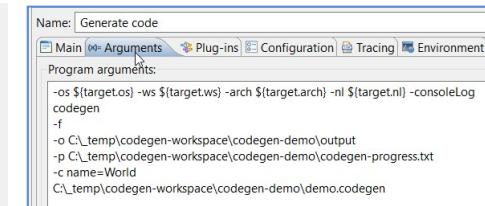
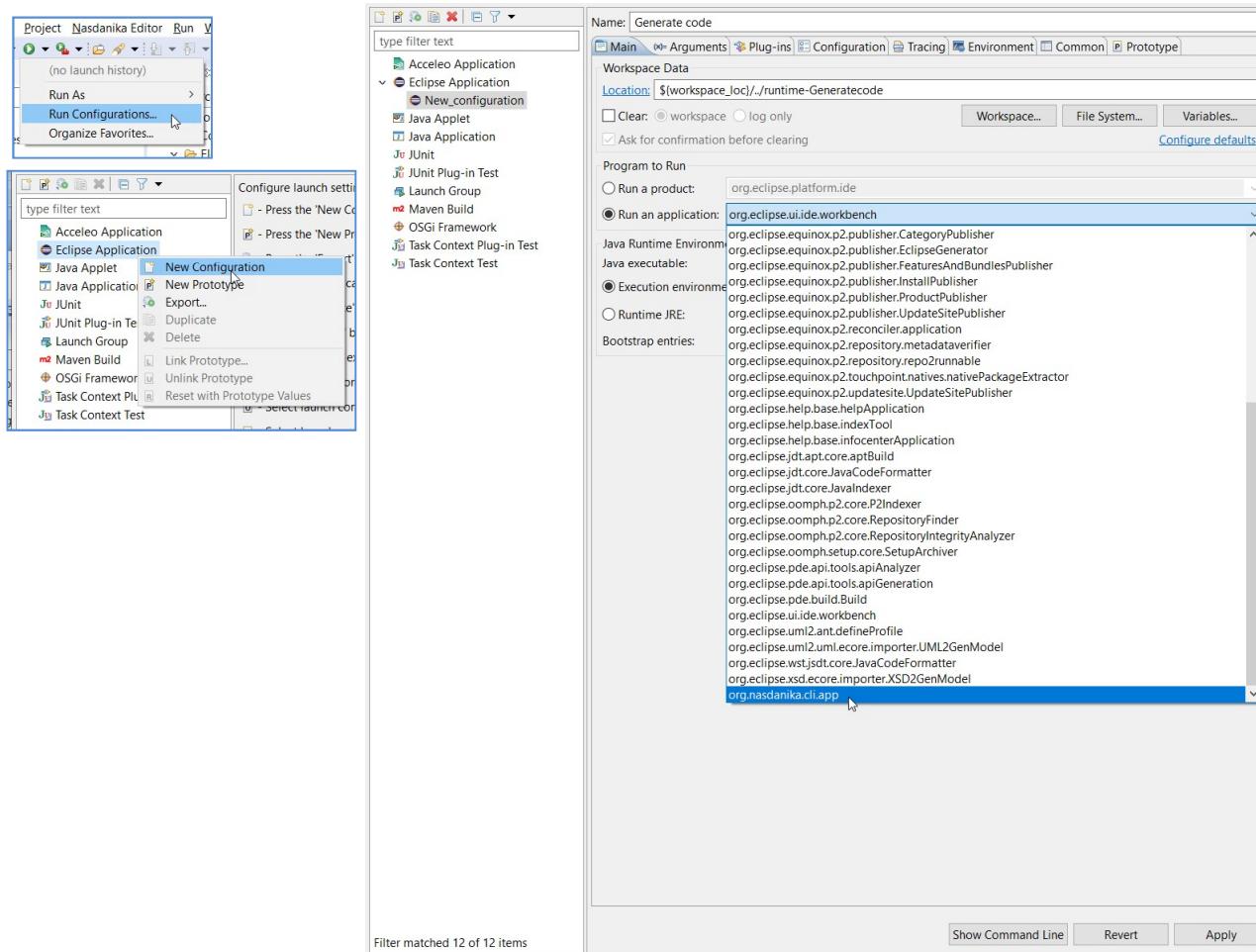
- Right click on model elements in the representation editor to add child elements
- Fill-out properties in the property view
- Ctrl-C/Ctrl-V to copy
- Drag-n-Drop to move from one element to another
- Right-Click > Load resource to reference model elements from external models

The screenshot shows the Eclipse Modeling Workbench interface. On the left, there is a 'Resource Set' view containing a 'Resource Group' node with children 'Configuration' and 'Elements'. A context menu is open over the 'Elements' node, with 'New Child' selected. A submenu is displayed, showing options like 'Container', 'File...', 'Resource Generator Adapter...', and 'Resource Generator Reference...'. In the center, there is a large empty workspace area. On the right, there is a 'Properties' view for the 'Resource Group' node. The 'Main' tab is selected, showing fields for 'Title' (empty), 'Id' (5f6ed16c-cafd-4b42-a84d-50c3750075da), and 'Enabled' (checked). The 'Code' tab is also visible. At the bottom of the interface, there is a 'Problems' view showing 0 items.

# Generation – IDE – Option 1: Eclipse Application



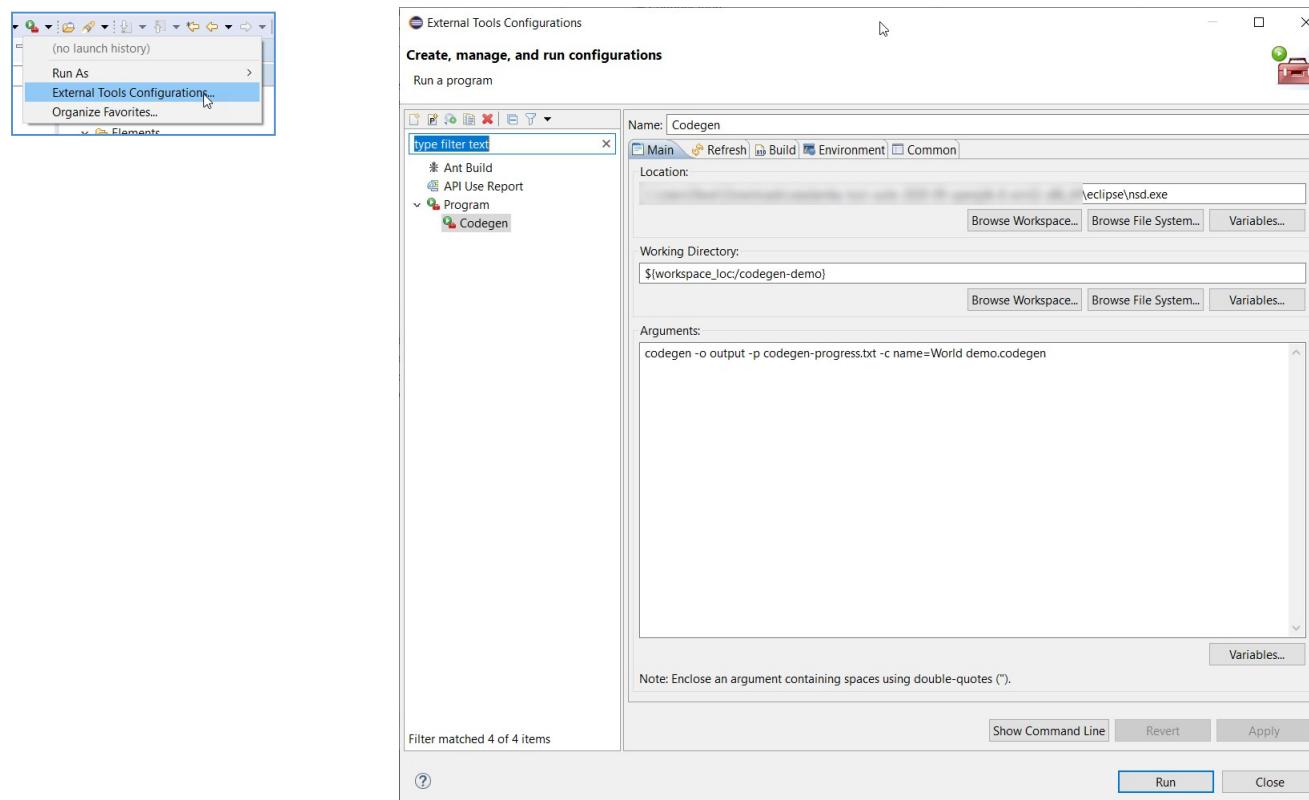
- Create a new Run configuration:
  - Eclipse application
  - Run an application: org.nasdanika.cli.app
  - Arguments according to the CLI documentation



# Generation – IDE – Option 2: External tool



- Create a new External tool configuration:
  - Program
  - Location – point to nsd.exe
  - Working directory – at your convenience,
    - *E.g. modeling project directory – shorter command line*
  - Arguments according to the CLI documentation



# Generation – output



- Look for the exit code – 0 is success
- Ignore net4j exception stack trace
- Explore progress output if needed to troubleshoot – very verbose

```
Problems Console <terminated> Codegen [Program] eclipse\nsd.exe (-1)
Exit code: 0
[WARN] Problem while deactivating CDOViewProviderRegistryImpl
java.lang.InterruptedException
    at java.util.concurrent.locks.AbstractQueuedSynchronizer.acquireSharedInterruptibly(AbstractQueuedSynchronizer.java:500)
    at java.util.concurrent.Semaphore.acquire(Semaphore.java:312)
    at org.eclipse.net4j.util.lifecycle.Lifecycle.lock(Lifecycle.java:310)
    at org.eclipse.net4j.util.lifecycle.Lifecycle.internalDeactivate(Lifecycle.java:118)
    at org.eclipse.net4j.util.lifecycle.ShareableLifecycle.internalDeactivate(ShareableLifecycle.java:52)
    at org.eclipse.net4j.util.lifecycle.Lifecycle.deactivate(Lifecycle.java:168)
    at org.eclipse.net4j.util.lifecycle.LifecycleUtil.deactivate(LifecycleUtil.java:235)
    at org.eclipse.net4j.util.lifecycle.LifecycleUtil.deactivate(LifecycleUtil.java:225)
    at org.eclipse.net4j.util.lifecycle.LifecycleUtil.deactivate(LifecycleUtil.java:251)
    at org.eclipse.emf.internal.cdo.bundle.Activator$Implementation.doStop(Activator.java:129)
    at org.eclipse.emf.internal.cdo.bundle.Activator$Implementation.stop(Activator.java:99)
    at org.eclipse.osgi.internal.framework.BundleContextImpl$4.run(BundleContextImpl.java:899)
    at org.eclipse.osgi.internal.framework.BundleContextImpl$4.run(BundleContextImpl.java:1)
    at java.security.AccessController.doPrivileged(Native Method)
    at org.eclipse.osgi.internal.framework.BundleContextImpl.stop(BundleContextImpl.java:891)
    at org.eclipse.osgi.internal.framework.EquinoxBundle.stopWorker0(EquinoxBundle.java:1029)
    at org.eclipse.osgi.internal.framework.EquinoxBundle$EquinoxModule.stopWorker(EquinoxBundle.java:370)
    at org.eclipse.osgi.container.Module.doStop(Module.java:658)
```

# Generation – CLI



- Remove `-Dpicocli.ansi=true` from `nsd.ini` if garbled output instead of color highlighting – no ANSI support
- Look for the exit code – 0 is success
- Ignore net4j stack trace

```
C:\Windows\system32\cmd.exe

C:\_temp\codegen-workspace\codegen-demo>F:\eclipse\nsd codegen -o output -p codegen-progress.txt -c name=World demo.codegen
Exit code: 0
[WARN] Problem while deactivating CDOViewProviderRegistryImpl
java.lang.InterruptedException
    at java.util.concurrent.locks.AbstractQueuedSynchronizer.acquireSharedInterruptibly(AbstractQueuedSynchronizer.java:1306)
    at java.util.concurrent.Semaphore.acquire(Semaphore.java:312)
    at org.eclipse.net4j.util.lifecycle.Lifecycle.lock(Lifecycle.java:310)
    at org.eclipse.net4j.util.lifecycle.Lifecycle.internalDeactivate(Lifecycle.java:118)
    at org.eclipse.net4j.util.lifecycle.ShareableLifecycle.internalDeactivate(ShareableLifecycle.java:52)
    at org.eclipse.net4j.util.lifecycle.Lifecycle.deactivate(Lifecycle.java:168)
    at org.eclipse.net4j.util.lifecycle.LifecycleUtil.deactivate(LifecycleUtil.java:235)
    at org.eclipse.net4j.util.lifecycle.LifecycleUtil.deactivate(LifecycleUtil.java:225)
    at org.eclipse.net4j.util.lifecycle.LifecycleUtil.deactivate(LifecycleUtil.java:251)
    at org.eclipse.emf.internal.cdo.bundle.Activator$Implementation.doStop(Activator.java:129)
    at org.eclipse.emf.internal.cdo.bundle.Activator$Implementation.stop(Activator.java:99)
    at org.eclipse.osgi.internal.framework.BundleContextImpl$4.run(BundleContextImpl.java:899)
    at org.eclipse.osgi.internal.framework.BundleContextImpl$4.run(BundleContextImpl.java:1)
    at java.security.AccessController.doPrivileged(Native Method)
    at org.eclipse.osgi.internal.framework.BundleContextImpl.stop(BundleContextImpl.java:891)
    at org.eclipse.osgi.internal.framework.EquinoxBundle.stopWorker0(EquinoxBundle.java:1029)
    at org.eclipse.osgi.internal.framework.EquinoxBundle$EquinoxModule.stopWorker(EquinoxBundle.java:370)
    at org.eclipse.osgi.container.Module.doStop(Module.java:658)
    at org.eclipse.osgi.container.Module.stop(Module.java:520)
    at org.eclipse.osgi.container.ModuleContainer$ContainerStartLevel.decStartLevel(ModuleContainer.java:1885)
    at org.eclipse.osgi.container.ModuleContainer$ContainerStartLevel.doContainerStartLevel(ModuleContainer.java:1760)
    at org.eclipse.osgi.container.SystemModule.stopWorker(SystemModule.java:275)
    at org.eclipse.osgi.internal.framework.EquinoxBundle$SystemBundle$EquinoxSystemModule.stopWorker(EquinoxBundle.java:202)
    at org.eclipse.osgi.container.Module.doStop(Module.java:658)
    at org.eclipse.osgi.container.Module.stop(Module.java:520)
    at org.eclipse.osgi.container.SystemModule.stop(SystemModule.java:207)
    at org.eclipse.osgi.internal.framework.EquinoxBundle$SystemBundle$EquinoxSystemModule$1.run(EquinoxBundle.java:220)
    at java.lang.Thread.run(Thread.java:748)

C:\_temp\codegen-workspace\codegen-demo>
```

# Server Mode



- Adjust `server.ini` if needed
  - Port number
  - Context path
  - Base URL for models, defaults to the current directory
- Start `server.exe`
- HTTP POST
  - Input data as payload – YAML or JSON
  - Zip archive as response
    - *output directory contains generation results*
    - *progress.txt file contains progress information*
- TODO – CURL demo

# Install Server as Windows Service



- TODO

# Modeling Process



- Manually create an example of desired generation output – a reference implementation
- Analyze resources in the reference implementation to identify variation points – things which may change between generations, e.g:
  - Maven coordinates – group and artifact ID's
  - Project name
  - Project description
  - Root Java package
- Identify needed generation inputs and how variation points values will be derived from them, e.g. root Java package may be derived from Maven group ID and artifact ID
- Create sample input or inputs, e.g. a YAML file
- Create model elements, generate with provided input, compare with the reference implementation

# Example



- Iteration over java/beans and then fields

The screenshot shows the Eclipse IDE interface with several open windows:

- Eclipse Maven Project**: Shows the project structure with a file named `eclipse-maven-project-context.xml`. It contains configuration for a Maven project with group: org.nasdanika and artifact: demo. The `beans` section defines a `Person` bean with fields: `FirstName`, `LastName`, and `Age`.
- Resource Set**: Shows the Maven project structure under `src/main/java`, including `Configuration`, `Elements`, and `Content` sections.
- Properties**: Shows the properties for the `Person` bean. The `Name` field is set to `$(name)`. The `Iterator` field is set to `java/beans`. The `Code` tab shows generated Java code for the `Person` class.
- Properties**: Shows the properties for the `Member Group Properties`. The `Title` is `Properties`, the `Id` is `7dda6199-1e3f-47d5-9c07-665719efce8d`, and the `Iterator` is set to `fields`.
- Parameters**: Shows the parameters for the `Person` constructor, which is set to `$(import/java.util.Map) data`.

```

Eclipse Maven Project  eclipse-maven-project-context.xml  Person.java
package org.nasdanika.demo;

import java.util.Map;

/**
 * <p></p>
 * <generated>
 */
public class Person {
    /**
     * @generated
     */
    private java.lang.String FirstName;

    /**
     * @generated
     */
    public java.lang.String getFirstName() {
        return FirstName;
    }

    /**
     * @generated
     */
    public void setFirstName(java.lang.String value) {
    }

    /**
     * @generated
     */
    private java.lang.String LastName;

    /**
     * @generated
     */
    public java.lang.String getLastname() {
        return LastName;
    }

    /**
     * @generated
     */
    public void setLastname(java.lang.String value) {
    }

    /**
     * @generated
     */
    private int Age;

    /**
     * @generated
     */
    public int getAge() {
        return Age;
    }

    /**
     * @generated
     */
    public void setAge(int value) {
    }

    /**
     * @generated
     */
    Person(Map data) {
    }
}

```