MODELIO OPEN SOURCE

Referencia:

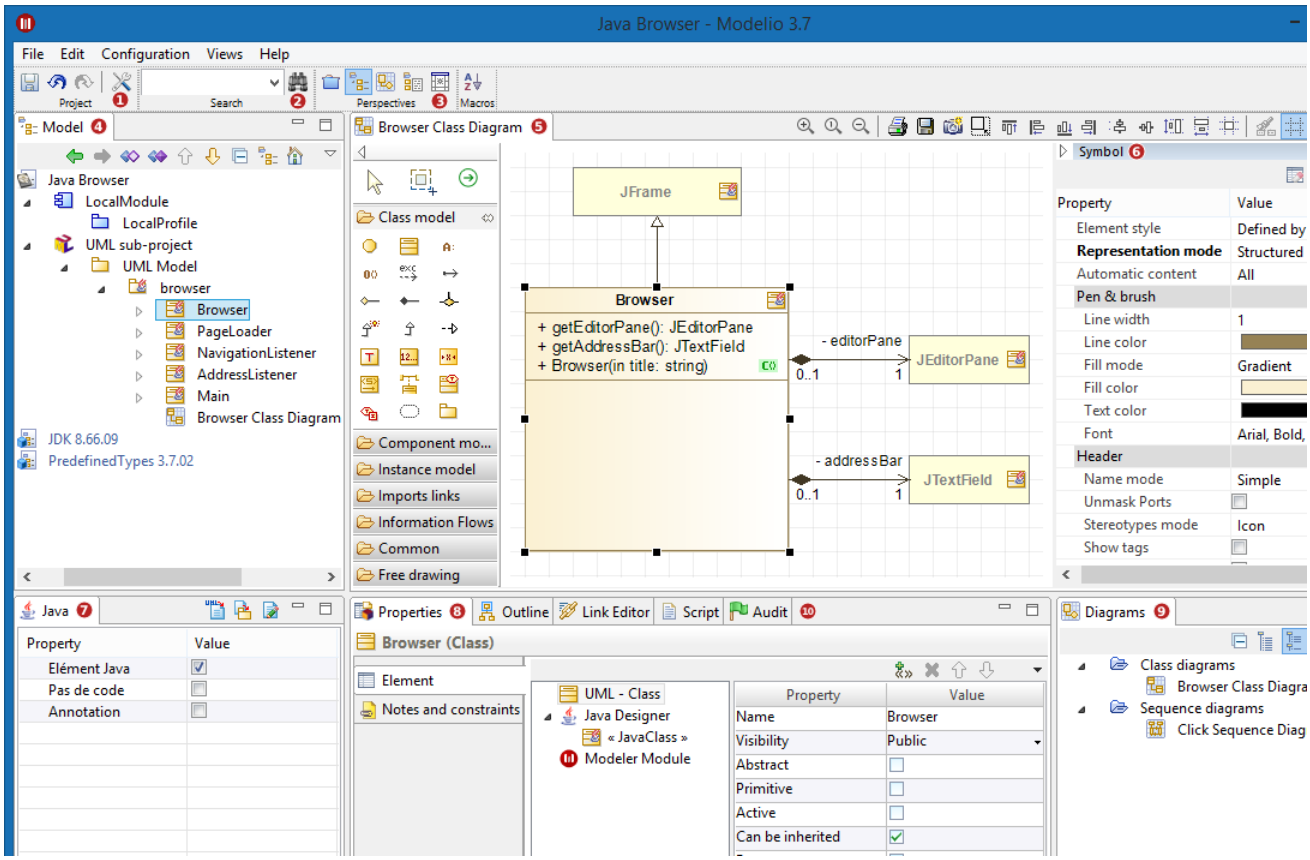
Descarga: https://www.modelio.org/

Documentación online: https://www.modelio.org/documentation-menu/user-manuals.html

INTERFAZ

Además de la barra de menús y la barra de herramientas con los accesos directos a las funciones principales, la interfaz de Modelio se compone de varias zonas bien diferenciadas:

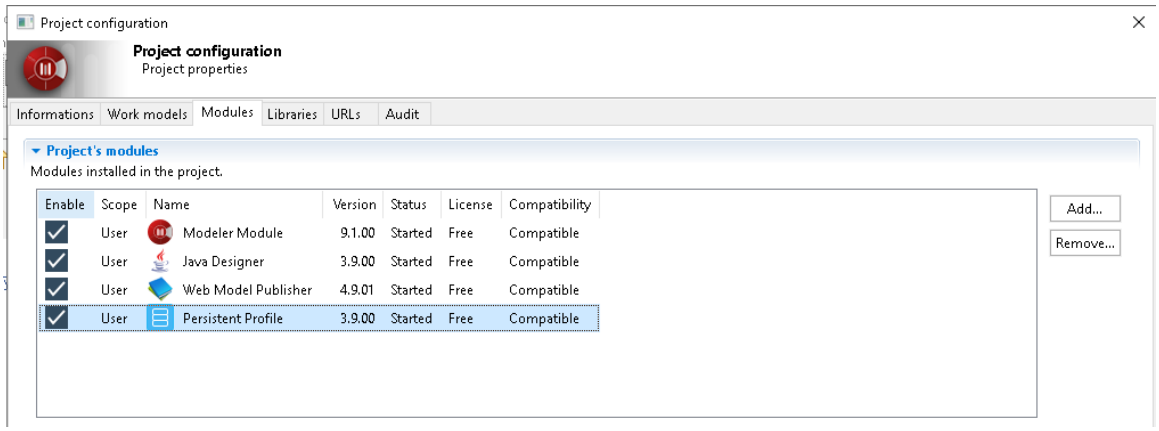
1. El configurador de proyectos . Esta herramienta te permite configurar la información de tu proyecto, modelos de trabajo, bibliotecas, módulos, auditoría y URLs.
2. La herramienta de búsqueda avanzada . Esta herramienta busca elementos basándose en un nombre parcial o completo.
3. Los botones de Perspectivas. Esta herramienta le permite cambiar entre perspectivas.
4. La vista Modelo . Aquí es donde puede ver, explorar y editar su modelo UML / BPMN.
5. La vista Edición de diagrama . Esta vista es donde puede ver y editar su modelo en forma de diagrama.
6. La vista Símbolo . Aquí es donde puede ver y editar las propiedades gráficas de sus diagramas.
7. La pestaña de módulos. Esta vista es donde puede ver y editar las opciones compradas por módulos (en este ejemplo, Java Designer).
8. La vista Propiedades . Aquí es donde puede agregar o eliminar estereotipos o valores etiquetados y ver y editar las propiedades UML de un elemento seleccionado en las vistas de navegación o edición.
9. La vista de diagramas . Aquí es donde puede ver, explorar y ordenar sus diagramas.
10. Otras vistas. Secuencia de comandos , editor de enlaces , auditoría , esquema ...



MODULOS

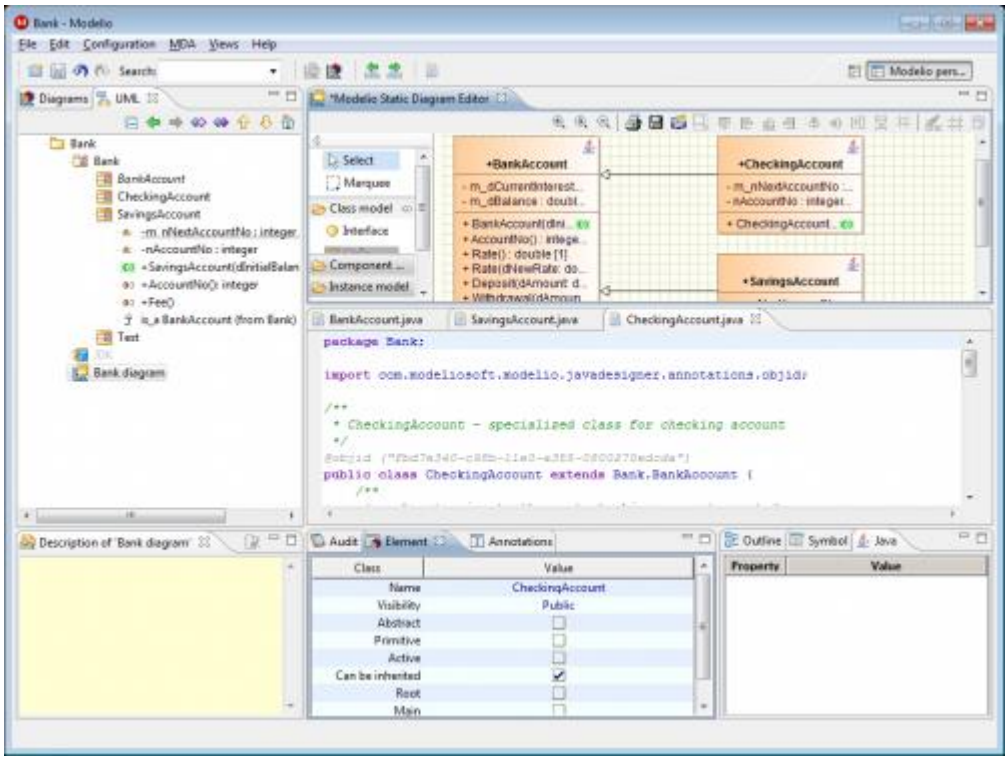
Los módulos amplían y adaptan Modelio Modeler proporcionando funcionalidades y servicios adicionales que satisfacen sus necesidades específicas. Los módulos de la Tienda Modelio los ofrece el creador de Modelio, así como los socios y la comunidad de código abierto de Modelio.

Para agregar o quitar un modulo de su proyecto debe ingresar en el menú Configuracion -> Modulos.



JAVA DESIGN MODULE

Proporciona generación de código Java y funcionalidades inversas. La arquitectura UML y el código Java se mantienen constantemente sincronizados. Es totalmente compatible con Java 7.



Modelio Java Designer genera una aplicación Java a partir de un modelo Modelio, y también gestiona la generación de su documentación con JavaDoc.

Modelio Java Designer proporciona cinco características principales:

- Generación de código Java
- Generación de documentación JavaDoc
- Compilación de archivos generados
- Reverso de las bibliotecas existentes
- Automatización de Java

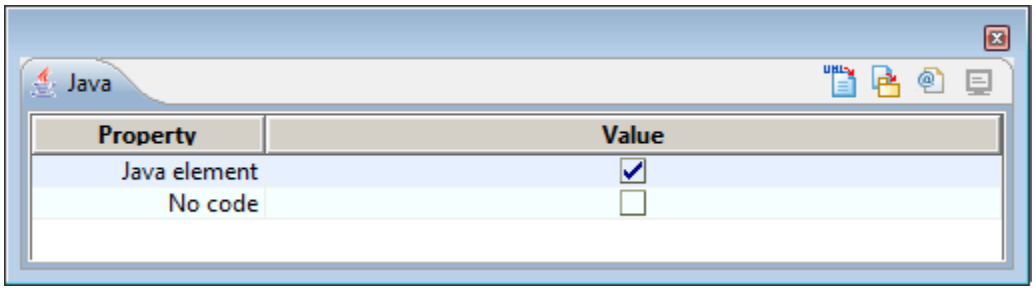
Modelio Java Designer proporciona dos modos de funcionamiento, para garantizar la coherencia en todo momento entre el modelo construido en Modelio y el código producido:

- El modo controlado por modelo genera la aplicación Java completa a partir del modelo y recupera el código insertado mediante marcadores.
- El modo de ida y vuelta combina la generación de código y las operaciones inversas, recupera el código escrito libremente y crea nuevos elementos de modelo cuando es necesario.

A continuación, se verá como activar y configurar la característica Java de acuerdo al elemento elegido: paquete, diagrama, clase, atributo, operación, asociación.

Vista Java Designer property en un paquete

Se utiliza para ingresar o modificar cierta información relevante para la generación de Java en el elemento seleccionado en Modelio, como notas , valores etiquetados o estereotipos .



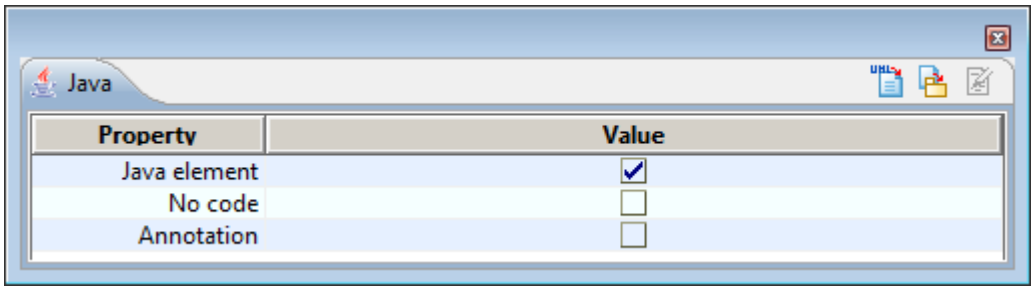
Claves:

- El campo "Java element " se utiliza para agregar el estereotipo << JavaPackage >>, que indica que el paquete lo maneja Java.
- El campo "No code" se utiliza para agregar el valor etiquetado {JavaNoCode}, lo que indica que el paquete no se generará.

Los iconos en la parte superior de la ventana se utilizan de la siguiente manera:

- Este icono se utiliza para iniciar la generación de Java.
- Este icono se utiliza para actualizar el modelo de las fuentes si es necesario.
- Este icono se utiliza para iniciar la generación de Javadoc.
- Este icono se utiliza para visualizar el Javadoc generado en un navegador web. En gris si no se genera el Javadoc.

Vista Java Designer property en una clase



Claves:

- El campo “Java element” se usa para agregar el estereotipo << JavaClass >>, que indica que la clase es manejada por Java.
- El campo "No code" se utiliza para agregar el valor etiquetado {JavaNoCode}, lo que indica que la clase no se generará.
- El campo "Anotation" se utiliza para especificar una anotación de Java en versiones viejas.

Los iconos en la parte superior de la ventana se utilizan de la siguiente manera:

- Este icono se utiliza para iniciar la generación de Java.
- Este icono se utiliza para editar el código Java generado.
- Este icono se utiliza para actualizar el modelo de las fuentes si es necesario.

Vista Java Designer property en un atributo

Claves:

- El campo "No code" se utiliza para agregar el valor etiquetado {JavaNoCode}, lo que indica que el atributo no se generará.
- La casilla de verificación "Propiedad Java" se utiliza para agregar el estereotipo << JavaAttributeProperty >>, lo que indica que el setter de anuncios getter creado para este atributo no se creará en el modelo, sino que se generará.
- El campo "Wrapper" se utiliza para especificar si se debe generar o no un tipo básico como clase contenedora. Disponible solo para tipos básicos con cardinalidad de 0..1.
- La casilla de verificación "Getter" indica si existe o no un getter actualizado automáticamente para este atributo.
- El cuadro combinado "Getter visibility" indica la visibilidad del setter, pero solo para las propiedades de Java. La opción "default" significa que se aplica la automatización regular para calcular esta visibilidad.
- La casilla de verificación "Setter" indica si existe o no un setter actualizado automáticamente para este atributo.
- El cuadro combinado "Setter visibility" indica la visibilidad del setter, pero solo para las Propiedades de Java. La opción "default" significa que se aplica la automatización regular para calcular esta visibilidad.
- La casilla de verificación "Static" indica que el atributo es estático.
- La casilla de verificación "Final" agrega el valor etiquetado {JavaFinal} al atributo e indica que es final.
- La casilla de verificación "Volátil" agrega el valor etiquetado {JavaVolatile} al atributo e indica que es volátil.
- La casilla de verificación "Transient" agrega el valor etiquetado {JavaTransient} al atributo e indica que es final.
- El campo "key" se utiliza para agregar el segundo parámetro al valor etiquetado {type}. Indica el tipo de clave que se utilizará para tipos de contenedores como HashMap.
- El campo "No initial value" se utiliza para deshabilitar la generación del valor inicial del atributo.
- El campo "template binding" se utiliza para especificar parámetros si el tipo de datos del atributo es un tipo genérico.

Los iconos en la parte superior de la ventana se utilizan de la siguiente manera:

 Este icono se utiliza para iniciar la generación de Java.
 Este icono se utiliza para editar el código Java generado.
 Este icono se utiliza para actualizar el modelo de las fuentes si es necesario.

Vista Java Designer property en una asociación

Claves:

- La casilla de verificación "No code " se utiliza para agregar el valor etiquetado {JavaNoCode}, lo que indica que el atributo no se generará.

- La casilla de verificación "Java property" se utiliza para agregar el estereotipo << JavaAssociationEndProperty >>, lo que indica que los getter y setter creados para esta asociación no se creará en el modelo, sino que se generará.
- La casilla de verificación "Getter" indica si existe o no un getter actualizado automáticamente para este atributo.
- El cuadro combinado "Getter visibility" indica la visibilidad del getter, pero solo para las propiedades de Java. La opción "Predeterminado" significa que se aplica la automatización regular para calcular esta visibilidad.
- La casilla de verificación "Setter" indica si existe o no un setter actualizado automáticamente para este atributo.
- El cuadro combinado "Visibilidad de Setter" indica la visibilidad del setter generado, pero solo para las Propiedades de Java. La opción "default" significa que se aplica la automatización regular para calcular esta visibilidad.
- La casilla de verificación "Static" indica que el atributo es estático.
- La casilla de verificación "Final" agrega el valor etiquetado {JavaFinal} al atributo e indica que es final.
- La casilla de verificación "Volátile" agrega el valor etiquetado {JavaVolatile} al atributo e indica que es volátil.
- La casilla de verificación "Transient" agrega el valor etiquetado {JavaTransient} al atributo e indica que es transitorio.
- El campo "key" se utiliza para agregar el segundo parámetro al valor etiquetado {Type}. Indica el tipo de clave que se utilizará para tipos de contenedores como HashMap.
- El campo "No init value" se utiliza para deshabilitar la generación del valor inicial del atributo.
- El campo "template binding" se utiliza para especificar parámetros si el tipo de datos del atributo es un tipo genérico.

Los iconos en la parte superior de la ventana se utilizan de la siguiente manera:

 Este icono se utiliza para iniciar la generación de Java.
 Este icono se utiliza para editar el código Java generado.
 Este icono se utiliza para actualizar el modelo de las fuentes si es necesario

La vista de propiedades de Java Designer en una operación

Claves:

- El campo "No code" se utiliza para agregar el valor etiquetado {JavaNoCode}, lo que indica que la operación no se generará.
- La casilla de verificación “Static” indica que la operación es de “clase”, es decir, compartida por todas sus instancias.
- La casilla de verificación "Sincronized" agrega el valor etiquetado {JavaSynchronize} a la operación e indica que la operación está sincronizada.
- La casilla de verificación "Native" agrega el valor etiquetado {JavaNative} a la operación e indica que la operación es nativa.

Los iconos en la parte superior de la ventana se utilizan de la siguiente manera:

 Este icono se utiliza para iniciar la generación de Java.
 Este icono se utiliza para editar el código Java generado.

Este icono se utiliza para actualizar el modelo de las fuentes si es necesario.

Vista Java Designer property en un parámetro

Claves:

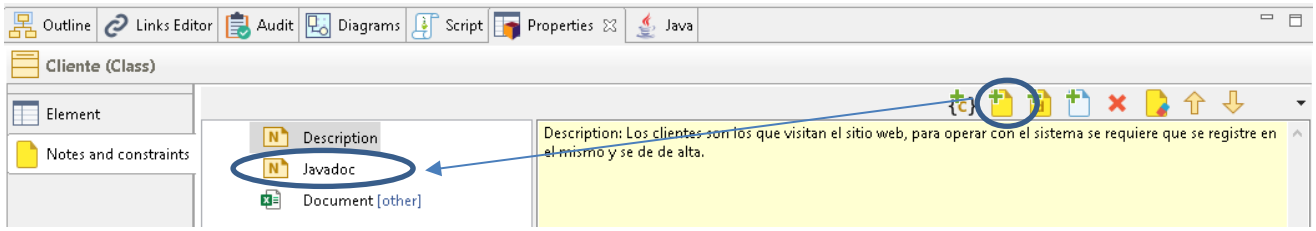
- El campo "Wrapper" se utiliza para determinar si se debe generar un tipo básico como clase contenedora.
- El campo "Collection" se utiliza para agregar el {tipo (x)}.
- El campo "Binding parameters for generic" se utiliza para especificar parámetros si el tipo de datos del atributo es un tipo genérico.

Los iconos en la parte superior de la ventana se utilizan de la siguiente manera:

Este icono se utiliza para iniciar la generación de Java.
Este icono se utiliza para editar el código Java generado.
Este icono se utiliza para actualizar el modelo desde las fuentes si es necesario.

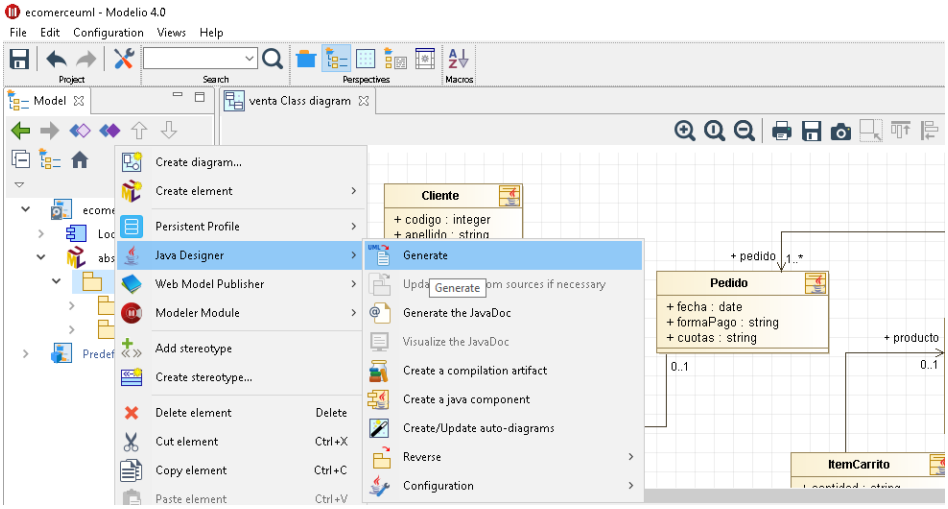
Especificando documentación con JavaDoc:

Modelio permite también agregar notas del tipo JavaDoc a todos los elementos del diagrama, de manera tal que el Modulo de Generación de Java tomará esta información y agregará la documentación especificada mediante las notas al código en los lugares correspondientes.



Ejecución del Generador de Código Java y documentación JavaDoc:

Una vez especificado todo lo necesario para la configuración del módulo, vamos a utilizar la opción de generar código eligiendo primeramente el paquete desde el cual se quiere generar código en el explorador del modelo. En la figura siguiente se muestra la opción a elegir.



Web Model Publisher Module

El módulo Web Model Publisher examina su modelo y produce documentación completa en HTML. Sus notas descriptivas se muestran con sus diagramas. Los hipervínculos se generan a partir del modelo, dentro de diagramas, capítulos y textos.

Un proyecto de alta calidad significa documentación completa, coherente, actualizada y relevante, pero ¿quién disfruta de la creación de documentos? ¿Y cómo puede garantizar que se respeten los estándares de documentación?

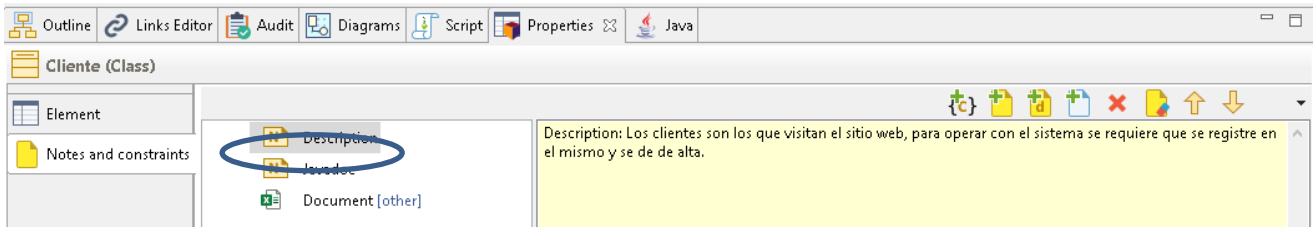
El editor de documentos Modelio es:

- Práctico : facilita la redacción y producción de documentación de calidad, simplemente aprovechando al máximo los modelos que crea.
- Intuitivo : Proporciona asistencia para la redacción de su documentación con el fin de obtener una calidad superior a la producción manual .
- Personalizable: un conjunto de plantillas de documentos predefinidas le da la opción de generar documentación para diferentes necesidades y definir sus propias plantillas de documentos.

Ejemplo: Puede generar documentación compuesta que incluye un glosario, requisitos, una sección de casos de uso, una sección de diagrama de clases y una matriz de trazabilidad , con enlaces de hipertexto incluidos sistemáticamente.

Para Personalizar la documentación que genera el modulo de documentación es necesario documentar todos los elementos de un diagrama. En este caso vamos a utilizar en la sección de propiedades la solapa “Notes and constraints” (notas y restricciones).

Nota: atributos y métodos de los diagramas de clases no serán documentados con la versión gratuita, sino es necesario comprar la modeliosoft que es la versión de modelio paga la cual tiene muchas mas opciones para configurar el proceso de generación de documentación.



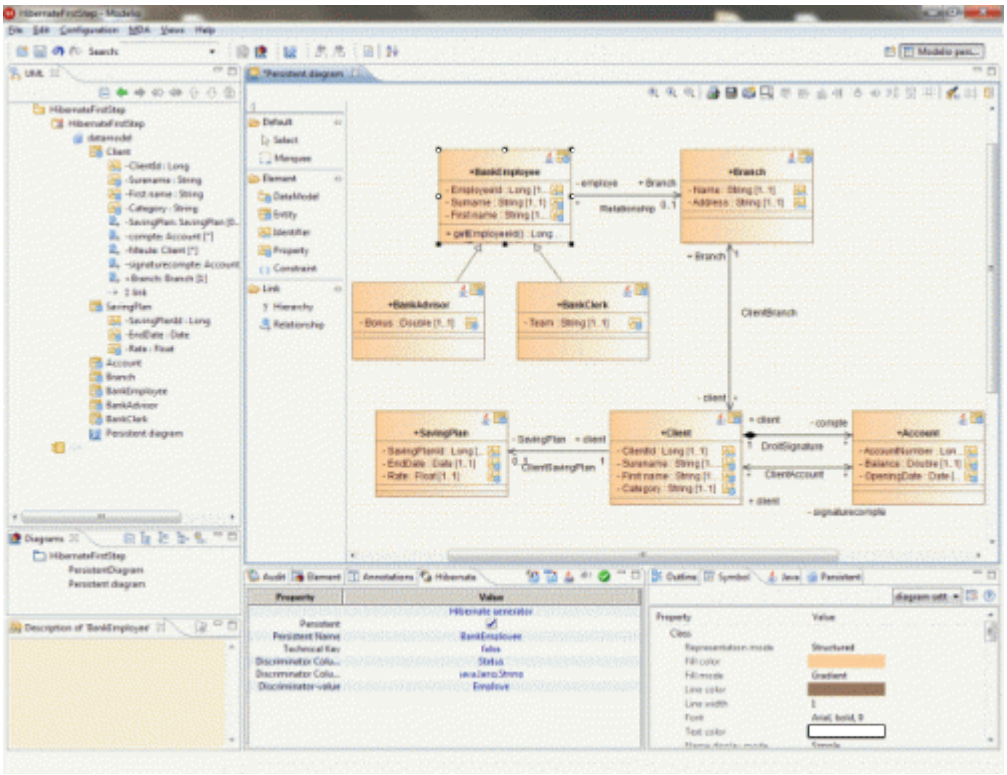
De la variedad de opciones de notas que podemos utilizar nos interesa la opción “Add description” que permitirá agregar una descripción a cada elemento y esta formará parte del resultado del proceso de la Generación de documentación con “Web Module Publisher”.

Abajo mostramos todas las opciones de documentación que provee modelio.

- Agregar restricción (s): abre la ventana "Agregar restricción", en la que puede seleccionar las restricciones que desea agregar al elemento seleccionado y luego mostrar su contenido en la zona de entrada / modificación.
- Agregar nota: abre la ventana "Agregar nota", en la que puede seleccionar las notas que desea agregar al elemento seleccionado y luego mostrar su contenido en la zona de entrada / modificación de notas.
- Agregar descripción: agrega directamente una nota de descripción al elemento
- Crea un documento adjunto con un contenido en la que puede definir el nombre, el formato MIME , luego puedes abrir el editor correspondiente.
- Eliminar: elimina la nota / restricción del elemento seleccionado.
- Limpiar nota / restricción: elimina el contenido de la nota o restricción seleccionada, pero no la nota o restricción en sí.
- Subir: mueve el elemento seleccionado hacia arriba en la estructura del modelo.
- Bajar: Mueve el elemento seleccionado hacia abajo en la estructura del modelo.

Hibérnate Designer Module

Es un módulo para modelar y automatizar la correspondencia entre Hibernate y la generación de clases Java. Admite el modelado de mapeo de Hibernate y genera el mapeo y las clases de Java que administran la persistencia de su aplicación.



Hibernate Designer trabaja en sincronía con el módulo Java Designer, para modelar y generar toda su aplicación de software.

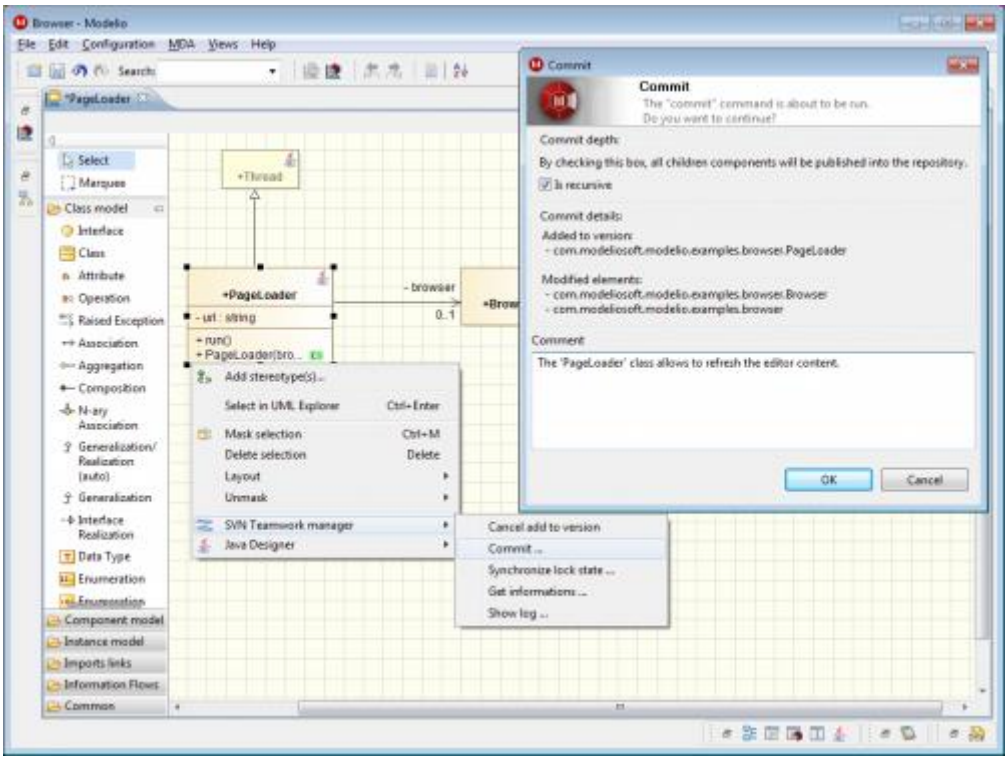
Nota: Hibernate Designer requiere el uso de un perfil persistente.

Team Work Manager Module

Proporciona un entorno de modelado UML colaborativo distribuido flexible integrado a Subversion. Cooperación distribuida en equipo a través de Internet o mediante redes locales.

Nota:

- Teamwork Manager requiere que se utilice la distribución Modeliosoft de Modelio.
- En Modelio 3, la función Teamwork Manager se proporciona con la entrega de Modelio Core (con versión comercial de Modelio solamente. No se proporciona con la versión de código abierto). Esta función necesita una licencia para ser utilizada después de su período de prueba.
- En Modelio 2 y versiones anteriores, Teamwork Manager es un módulo de Modeliosoft.


Browser - Modelio
File Edit Configuration MDA Views Help
Search
*PageLoader
Select
Marquee
Class model
Interface
Class
Attribute
Operation
Raised Exception
Association
Aggregation
Composition
N-ary Association
Generalization/ Realization (auto)
Generalization
Interface Realization
Data Type
Enumeration
Component model
Instance model
Imports links
Information Flows
Common
+Thread
+PageLoader
- url : string
+ run()
+ PageLoader(bro...
- browser
0..1
Add stereotype(s)...
Select in UML Explorer Ctrl+Enter
Mask selection Ctrl+M
Delete selection Delete
Layout
Unmask
SVN Teamwork manager
Java Designer
Cancel add to version
Commit ...
Synchronize lock state ...
Get informations ...
Show log ...
Commit
Commit
The "commit" command is about to be run.
Do you want to continue?
Commit depth:
By checking this box, all children components will be published into the repository.
Is recursive
Commit details:
Added to version:
- com.modeliosoft.modelio.examples.browser.PageLoader
Modified elements:
- com.modeliosoft.modelio.examples.browser.Browser
- com.modeliosoft.modelio.examples.browser
Comment
The 'PageLoader' class allows to refresh the editor content.
OK
Cancel