



Diseño de interfaz de usuario

Fco. Javier Bohórquez Ogalla

Índice

1. Visión general	3
1.1. Intérprete	3
1.2. runTree	4
1.3. Sitio web	5
1.3.1. Home	5
1.3.2. Sobre OMI	6
1.3.3. Contacto	6
1.3.4. Índice de la documentación	7
1.3.5. Documento	7
1.3.6. Navegador de gramática	8
1.3.7. Navegador de clases	8
1.3.8. Navegador de ficheros	9

1. Visión general

En esta sección se describe el diseño de la interfaz del usuario a partir del análisis llevado a cabo.

El intérprete presenta una interfaz de consola de comandos, por lo que el diseño se corresponde con una descripción de los usos del comando y el listado de las opciones que acepta.

El cliente runTree y la web en general presenta una interfaz web. El diseño se corresponde con una descripción gráfica de las páginas que la conforman y de las secciones que las componen.

1.1. Intérprete

El intérprete es un programa de consola de comando por lo que la interfaz con el usuario no es gráfica.

El comando “omi” permite ejecutar el intérprete. Se puede usar de las siguientes forma:

- omi [opciones] < *fichero* > [argumentos...]
- omi -c < *codigo* > [argumentos...]
- omi -i [argumentos...]
- omi -sj < *fichero.json* >

El listado de opciones que acepta el comando a continuación:

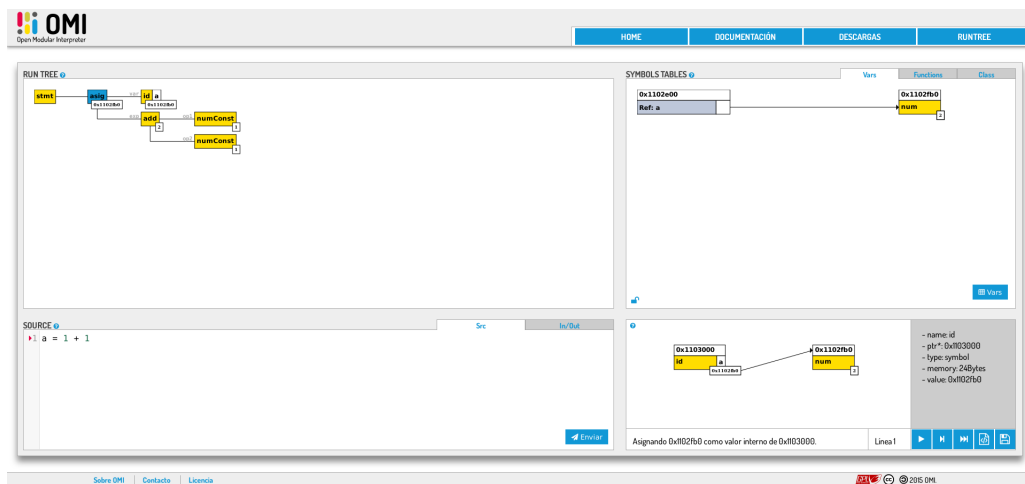
- i : Ejecuta el intérprete de forma interactiva.
- c < *codigo* > : Interpreta el código dado.
- l < *fichero.so* > : Carga el módulo < *fichero.so* >.
- h : Muestra la ayuda.
- V : Muestra la versión.
- j < *fichero.json* > : Imprime una descripción del procesos en formato json en el fichero < *fichero.json* >.

- x < pasos > : Obtiene < pasos > pasos del proceso de interpretación en cada petición.
- s : Ejecuta como servidor en el puerto 8888.

1.2. runTree

runTree es un cliente web del intérprete OMI. Muestra información sobre el proceso de interpretación y permite navegar por este.

El sistema lo compone una única página web con un diseño de cuadrícula.



La primera parte de la retícula, superior izquierda, muestra el árbol sintáctico correspondiente al código enviado. En este árbol se marcarán los nodos a medida que se van ejecutando y resolviendo semánticamente. Además mostrará información directa de cada nodo tal como el nombre, el valor o el tipo.

En la segunda parte, superior derecha, muestra las tablas de símbolos de variables, funciones y clases. Esta sección permite navegar por las tablas de símbolos y los elementos que serán referenciados desde las mismas.

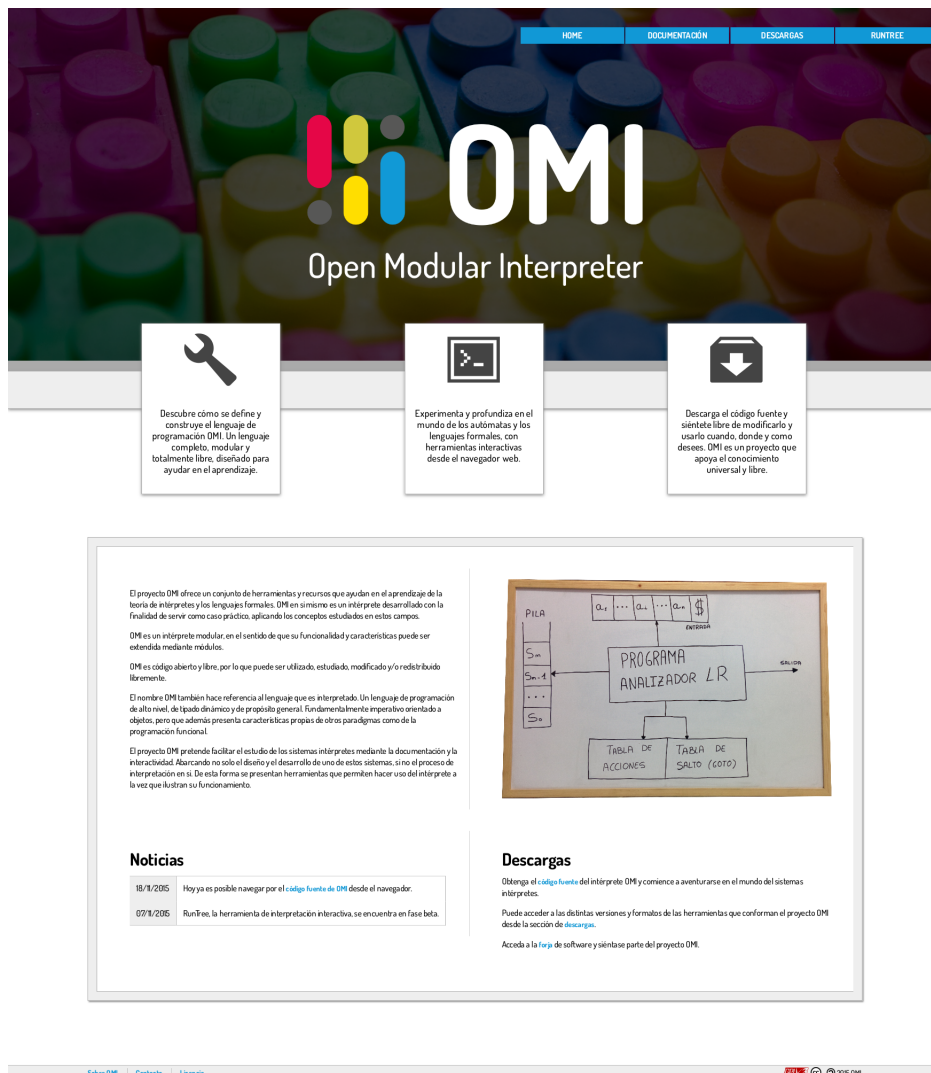
En la tercera parte, inferior izquierda, se muestra el código fuente y la interfaz de entrada/salida del programa.

En la última sección, inferior derecha, se muestra una consola informativa, en la que aparece una descripción del nodo actualmente en ejecución o los nodos seleccionados por el usuario. También se colocan en esta sección las opciones de control para avanzar un paso, una sentencia, la reproducción automática, abrir un fichero local o guardar el código en un fichero local.


1.3. Sitio web

Según el análisis de la interfaz de usuario llevado a cabo se procede a detallar gráficamente cada página descrita.

1.3.1. Home



1.3.2. Sobre OMI



Open Modular Interpreter

HOME


DOCUMENTACIÓN

DESCARGAS

RUNTIME

Sobre OMI

El proyecto OMI tiene como razón de ser servir como apoyo en el aprendizaje de los conceptos detrás de los sistemas intérpretes. Se vale de la teoría de autómatas y los lenguajes formales para construir una serie de herramientas que ayudan en el proceso de estudio de estos sistemas.



Motivación

La teoría de autómatas y los lenguajes formales son la base para la construcción de sistemas intérpretes. Muchos de los conceptos estudiados en estos campos sirven para construir otras ramas de estudio como la teoría de intérpretes y compiladores.

Normalmente, por limitaciones de tiempo, los cursos académicos relacionados con el estudio de estos campos están enfocados a los conceptos teóricos. Llevándose a cabo tan solo algunas prácticas que refuerzan el aprendizaje y ayudan a ver la aplicación real mediante casos sencillos y simplificados. Sería de gran ayuda en este proceso disponer de un caso práctico completo, que ilustre cómo se define y construye un lenguaje de programación actual, y que pueda ser consultado cuando, donde y por quien lo desee. OMI pretende ser una fuente de información que complemente el estudio práctico en estas áreas.

Aunque existen herramientas que dan soporte a la construcción de sistemas intérpretes, es difícil encontrar alguna que ayude a comprender cómo estos se construyen y funcionan. Los intérpretes modernos no están enfocados en ilustrar la tarea que llevan a cabo, tienen como propósito ejecutar programas de forma óptima y efectiva. Sería de utilidad disponer de una herramienta que documente los procesos sintácticos y semánticos llevados a cabo durante la interpretación de código fuente. OMI tiene como propósito generar información relativa al proceso de interpretación que muestre el funcionamiento del intérprete.

La comunidad académica no solo precisa de un objeto de estudio, si no también de un objeto que ayude a formar la comunidad. El proyecto OMI pretende ser una base para crear una comunidad centrada en el estudio de los lenguajes formales, en las que todos puedan contribuir a la vez de beneficiarse.

Autoría

El proyecto OMI ha sido diseñado, desarrollado y promovido por Fco. Javier Bohórquez Ogalla. Alumno de ingeniería técnica en informática y profundo entusiasta de las nuevas tecnologías. Con un gran interés personal y profesional en los lenguajes formales, y en las teorías y conceptos detrás de estos.


OMI ha seguido el camino marcado por la tutoría de Iván Ruiz Rubio. Profesor de la Universidad de Cádiz. Su dedicación y criterio han sido una pieza clave para el desarrollo de este proyecto.

Por el trabajo y dedicación invertidos, así como por el apoyo dado, merece mención especial Sandra Fernández Domínguez. Sin conocimientos ni formación anterior en la materia ha hecho todo lo posible para que este proyecto sea una realidad.


Por su colaboración en la elaboración del logotipo y su desempeño en consultoría de diseño, se merece una mención especial Luis Manuel Vaca.

Organismos


OMI ha sido desarrollado como proyecto fin de carrera para la Universidad de Cádiz.



[Sobre OMI](#) | [Contacto](#) | [Licencia](#)

 © 2015 OMI

1.3.3. Contacto



Open Modular Interpreter

HOME


DOCUMENTACIÓN

DESCARGAS

RUNTIME

Contacto

OMI es lo que es su comunidad. El proyecto espera evolucionar y crecer para convertirse en una pieza clave para el aprendizaje de los lenguajes formales. Su comunidad es parte esencial de este proceso.



Para cualquier cuestión relacionada con el proyecto OMI no dude en ponerse en contacto con nuestro equipo en contacto@omi-project.com.

Si desea colaborar con el desarrollo del proyecto por favor póngase en contacto con el equipo de OMI en colaboradores@omi-project.com.


Para el reporte de errores en algunas de las piezas software del proyecto puede mandar un correo electrónico a bugs@omi-project.com.

Cualquier cuestión sobre la licencia de uso puede ser tratada en licencia@omi-project.com.

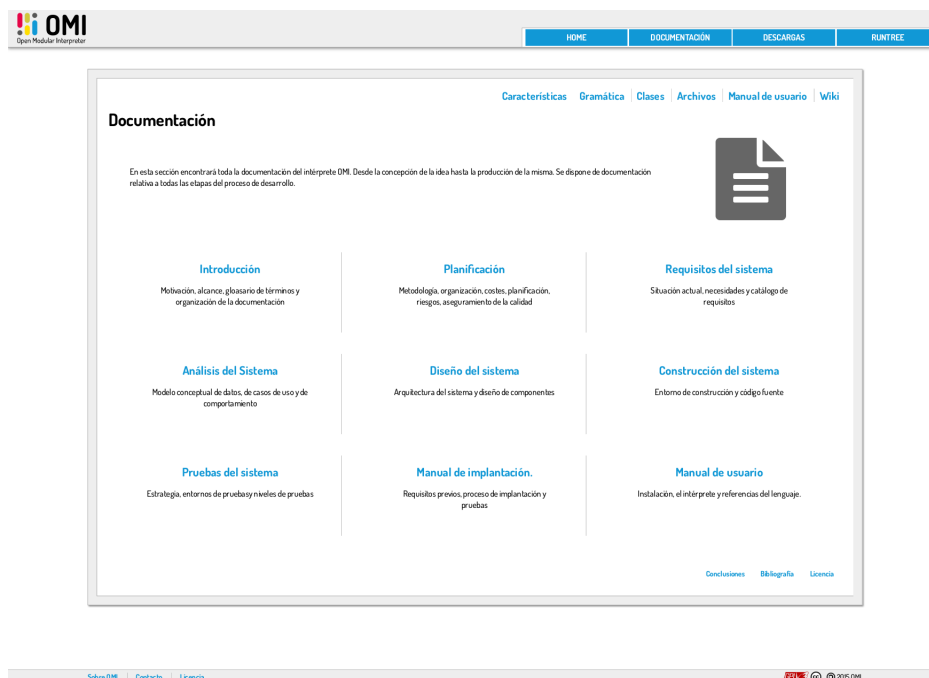
Para contactar directamente con el promotor del proyecto puede escribir a fbohorquez@omi-project.com.

También puede suscribirse a la lista de correo del proyecto en lista@omi-project.com.

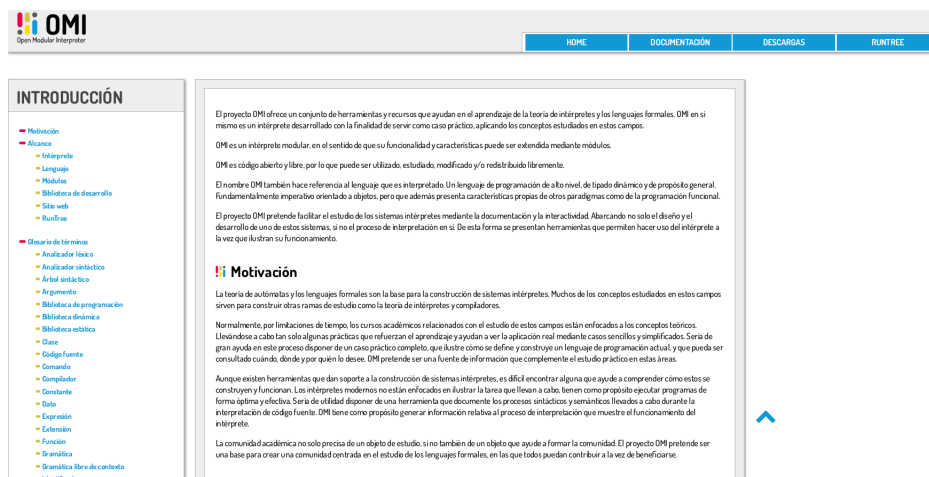
[Sobre OMI](#) | [Contacto](#) | [Licencia](#)

 © 2015 OMI


1.3.4. Índice de la documentación



1.3.5. Documento



1.3.6. Navegador de gramática

 OMI
Open Modular Interpreter

HOME	DOCUMENTACIÓN	DESCARGAS	RUNTREE
------	---------------	-----------	---------

Gramática

program:

```
program ::= stats
        | empty
```

no references


stats:

```
stats ::= stmt ';'
       | stmt ';' stats
       | stmts
       | label stmts
```

referenced by:

- [stmts_stmts](#)
- [declarator](#)
- [statement](#)
- [int](#)
- [function](#)
- [function_declarator](#)
- [function_argument](#)
- [generalizer](#)
- [if](#)
- [loop](#)
- [assignment](#)
- [return](#)
- [break](#)
- [while](#)
- [do](#)

1.3.7. Navegador de clases



Open Modular Interoperability

HOME

DOCUMENTATION

DESCARGAS

RUNTIME


Clases
 Índice de clases
 Jarramigo de la clase
 Miembros de las clases

INDICE DE CLASES

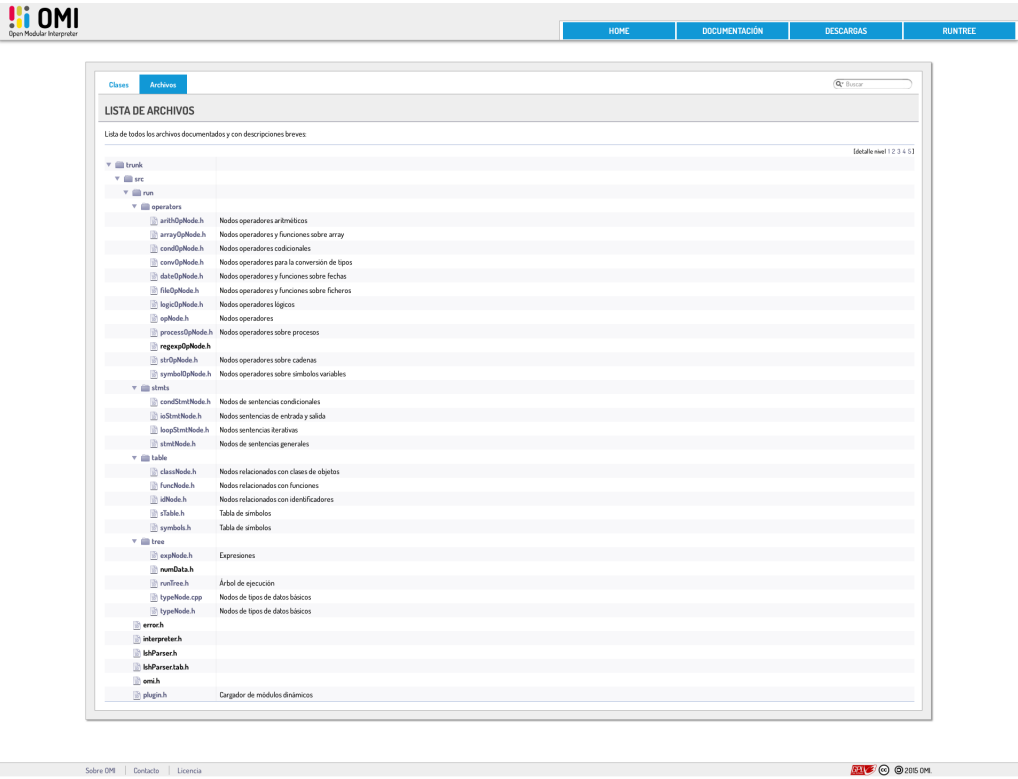
<div>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z _</div>			
<div>P</div> PluginReader <div>a</div> addNode andNode arrayChildNode arrayConstNode arrayDeleteNode arrayFirstNode arrayInsertNode arrayLastNode arrayNode arrayPathNode arrayRepeatNode arraysOfNode arraysOfIntNode arraysOfStringNode assignNode assignRefNode <div>b</div> boolConstNode boolConnNode boolNode breakException breakNode <div>c</div> caseNode catNode classNode copyNodeNode contextFunction continueException continueNode <div>d</div> dataNode	dateTimeNode decimalNode deleteNode dirNode doubleNode <div>e</div> emptyNode eqNode evalNode execNode exitNode exitProcessNode expandNode expNode <div>f</div> fahrenheitNode fcolorNode feapNode firstNode floatNode findNode floatConnNode forecastElementNode forecastNode forkNode forNode freqNode freashNode freekNode ftidNode functionCallNode functionGetNode functionNode funtioNode <div>g</div> geoNode getClassNode getElemNode getNode	getNode getNode_arrayValue getNode_atInterface getIdNode getIpNode getPathNode getPortNode globusNode gotoException graphNode gridNode <div>i</div> idNode idNodeNode idNodeNode ifNode ifgraphNode implNodeNode incargNode inclogNode includeNode inputNode insertConnNode interpreter isarrayNode isBaseNode isNullNode isNumNode isObjectNode isStringNode isSymTable <div>j</div> jobNode logNode logNode logNode lowerNode libNode <div>m</div> mapNode	multiNode multiNode <div>n</div> namespaceNode netNode newNode newargNode nextNode nilNodeNode noNode noNode noNodeNode numConstNode numNode <div>o</div> objectNode objectTypeNode orNode <div>p</div> parentNode partialFunctionNode pluginNode powNode printableNode printNode privateNode processNode prodNode <div>r</div> reduceNode refNode refParamNode regexConstNode regexNode replaceNode returnException returnNode runNode <div>s</div> searchNode signalHandlerNode signalNode sizeNode sizeOfNode sleepNode splitNode sqrtNode stable staticNode strNode strConstNode streamNode stringNode strNode subNode substituteNode symbolTable <div>t</div> termNode thisNode throwException threeNode timeNode tryNode typedNode <div>u</div> upperNode <div>v</div> valNode whileNode withNode

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z _

Schur DSE
Contributors
Licencia

 CC BY-SA 4.0

1.3.8. Navegador de ficheros



1.3.9. Descargas

