Documentation
for the
Eiffel Backend Service

Compilation
Execution
Flat View
Contract View
Descendants
Possible Errors
Using the Service

Compilation

http://localhost:9090/compile?clean=true;path="your_path";id=""
The first time you use a project, you have to provide a non-empty path and a blank id, and an optional clean compile flag.
The server will create a new project directory for you and return the unique id in the "id" key of the response object.
The response object JSON for compile looks something like this:


```
[{
	"Compile_Message":"Eiffel Compilation Manager\nVersion 14.05.9.5158 GPL Edition -
win64\n\nDegree 6: Examining System\nDegree 5: Parsing Classes\nDegree 4: Analyzing
Inheritance\nDegree 3: Checking Types\nDegree 2: Generating Byte Code\nFreezing System
Changes\nDegree -1: Generating Code\nSystem Recompiled.",
	"Output_Message":"",
	"Error_Message":"",
	"Syntax_Message":"",
	"Warning_Message":"Warning code: Unused_local_warning\n\nWarning: unreferenced local
variable(s)\nWhat to do: Remove it if you don't plan to use it in the future.\n\nClass:
APPLICATION\nFeature: make\nUnused local is: \n\tc: INTEGER_32\n\n--------------------------------
------------------------------------------------",
	"Dump_Message":"Eiffel Compilation Manager\nVersion 14.05.9.5158 GPL Edition -
win64\n\nDegree 6: Examining System\nDegree 5: Parsing Classes\nDegree 4: Analyzing
Inheritance\nDegree 3: Checking Types\nDegree 2: Generating Byte Code\nFreezing System
Changes\nDegree -1: Generating Code\n----------------------------------------------------------------
------------------\n\nWarning code: Unused_local_warning\n\nWarning: unreferenced local
variable(s)\nWhat to do: Remove it if you don't plan to use it in the future.\n\nClass:
APPLICATION\nFeature: make\nUnused local is: \n\tc: INTEGER_32\n\n--------------------------------
------------------------------------------------\nSystem Recompiled",
	"Error":null,
	"Warning":[{
		"Warning_Code":"Unused_local_warning\n",
		"Warning":"unreferenced local variable(s)\n",
		"What_to_do":"Remove it if you don't plan to use it in the future.\n\n",
		"Class":"APPLICATION\n",
		"Feature":"make\n",
		"After_Feature":"Unused local is: \n\tc: INTEGER_32\n\n",
		"Dump":"Warning code: Unused_local_warning\n\nWarning: unreferenced local
	variable(s)\nWhat to do: Remove it if you don't plan to use it in the future.\n\nClass:
	APPLICATION\nFeature: make\nUnused local is: \n\tc: INTEGER_32\n\n"}]
}]
```


Compile_Message gives the compile message
Output_Message gives the C compilation results and any other output messages if any
Error_Message gives the unparsed error message (excluding syntax error)
Syntax_Message gives only the unparsed syntax error message
Warning_Message gives the unparsed warning message
Dump_Message gives the whole compile dump message with errors, warnings, and the compile results
Error gives the parsed JSON object for the error message with its keys
Warning gives the parsed JSON object for the warning message with its keys
	Warning Code, Warning, What_to_do, Class, Feature, After_Feature(the part after the feature that does not have any name), and a dump of the whole warning.

An Example of an error message


```
[{
	"Compile_Message":"Eiffel Com",
	"Output_Message":"",
	"Error_Message":"Error code: VEEN\n\nError: unknown identifier.\nWhat to do: make sure
that identifier, if needed, is final name of\n  feature of class, or local entity or formal
argument of routine.\n\nClass: ACCOUNT\nFeature: make\nIdentifier: afgnh\nTarget type: [like
Current] attached ACCOUNT\nLine: 19\n      ensure\n->      bal_set: balance=afgnh\n      end\n\n-
------------------------------------------------------------------------------",
	"Syntax_Message":"",
	"Warning_Message":"",
	"Dump_Message":"Eiffel Compilation Manager\nVersion 14.05.9.5158 GPL Edition -
win64\n\nDegree 6: Examining System\nDegree 5: Parsing Classes\nDegree 4: Analyzing
Inheritance\nDegree 3: Checking Types\n-------------------------------------------------------------
---------------------\n\nError code: VEEN\n\nError: unknown identifier.\nWhat to do: make sure
that identifier, if needed, is final name of\n  feature of class, or local entity or formal
argument of routine.\n\nClass: ACCOUNT\nFeature: make\nIdentifier: afgnh\nTarget type: [like
Current] attached ACCOUNT\nLine: 19\n      ensure\n->      bal_set: balance=afgnh\n      end\n\n-
------------------------------------------------------------------------------\n",
	"Error":[{
		"Error_Code":"VEEN\n",
		"Error":"unknown identifier.\n",
		"What_to_do":"make sure that identifier, if needed, is final name of\n  feature of
	class, or local entity or formal argument of routine.\n\n",
		"Class":"ACCOUNT\n",
		"Feature":"make\n",
		"Line":19,
		"Before_Line":"Identifier: afgnh\nTarget type: [like Current] attached ACCOUNT\n",
		"After_Line":"      ensure\n->      bal_set: balance=afgnh\n      end\n\n",
		"Dump":"Error code: VEEN\n\nError: unknown identifier.\nWhat to do: make sure that
		identifier, if needed, is final name of\n  feature of class, or local entity or
		formal argument of routine.\n\nClass: ACCOUNT\nFeature: make\nIdentifier:
		afgnh\nTarget type: [like Current] attached ACCOUNT\nLine: 19\n      ensure\n->
		bal_set: balance=afgnh\n      end\n\n"}],
	"Warning":null}]
```


The Error has the following keys

Error_code, error, what_to_do, class, feature, line, before_line (part between feature and line) , after_line(part after line that does not have a name) and the dump of the whole error

For further compile calls, you don't need to provide the path, simply provide the "id" and the optional clean compile flag.

Note 1 : After the server has started, it needs to set project directory and location once for any new project(by the path flag in the compile parameters). If however the server is terminated while working, it will need the path location again for the project it was working on. So, the server must run continuously.

Note 2 : The server should always be started before the project is opened (i.e. compiled, otherwise it is giving me you don't have the permission to write to its EIFGENS's error)

Execution

http://localhost:9090/run?id=”id-given”

When you run a project, you must always supply the project id in the URL. The server already has the path information stored with it so it does not need it.

The response object JSON for error free execution looks something like this:


```
[{
	"Execution_Output":"Hello Eiffel World!\nNumber not out of range\n0",
	"Error_Message":"",
	"Syntax_Message":"",
	"Warning_Message":"",
	"Compile_Errors":null,
	"Warnings":null,
	"Runtime_Errors":null
}]
```


Execution_Output gives the output of the execution
Error_Message gives the unparsed compile error message (excluding syntax error) and if there is a runtime error, it gives the unparsed runtime error message.
Syntax_Message gives only the unparsed syntax error message
Warning_Message gives the unparsed warning message
Compile_Errors is the parsed JSON object for compile time error. Its format is the same as the one in compilation
Warning is the parsed JSON object for warnings. Its format is the same as the one in compilation.
Runtime_Errors is the parsed JSON array for the runtime errors.

An Example of a run time error message


```
[{
	"Execution_Output":"Hello Eiffel World!\nNumber not out of range\n0",
	"Error_Message":"\nsample: system execution failed.\nFollowing is the set of recorded
exceptions:\n\n---------------------------------------------------------------------------------
\nClass / Object	Routine		Nature of exception	Effect\n--------------
-----------------------------------------------------------\nACCOUNT		_invariant
@3	balance_positive:	\n<0000004625A17A68>		Class
invariant violated.	Fail\n---------------------------------------------------------------------
------------\nACCOUNT		_invariant
\n<0000004625A17A68>		Routine failure.	Fail\n----------------
-----------------------------------------------------------\nACCOUNT		withdraw @3
\n<0000004625A17A68>		Routine failure.	Fail\n----------------
-----------------------------------------------------------\nAPPLICATION	make @5
\n<0000004625A175A8>		Routine failure.	Fail\n----------------
-----------------------------------------------------------\nAPPLICATION	root's
creation		\n<0000004625A175A8>		Routine
failure.	Exit\n---------------------------------------------------------------------
----------\n",
	"Syntax_Message":"",
	"Warning_Message":"Warning code: Unused_local_warning\n\nWarning: unreferenced local
variable(s)\nWhat to do: Remove it if you don't plan to use it in the future.\n\nClass:
APPLICATION\nFeature: make\nUnused local is: \n\tc: INTEGER_32\n\n------------------------------
-----------------------------------------------",
	"Compile_Errors":null,
	"Warnings":[{
```

```
                "Warning_Code":"Unused_local_warning\n",
                "Warning":"unreferenced local variable(s)\n",
                "What_to_do":"Remove it if you don't plan to use it in the future.\n\n",
                "Class":"APPLICATION\n",
                "Feature":"make\n",
                "After_Feature":"Unused local is: \n\tc: INTEGER_32\n\n",
                "Dump":"Warning code: Unused_local_warning\n\nWarning: unreferenced local
        variable(s)\nWhat to do: Remove it if you don't plan to use it in the future.\n\nClass:
        APPLICATION\nFeature: make\nUnused local is: \n\tc: INTEGER_32\n\n"}],

        "Runtime_Errors":[{
                "Class":"ACCOUNT<0000004625A17A68>",
                "Feature":"_invariant",
                "Routine":"3",
                "Message":"balance_positive:Class invariant violated.",
                "Effect":"Fail",
                "Initial_Text":"\nsample: system execution failed.\nFollowing is the set of
        recorded exceptions:\n"},
                {"Class":"ACCOUNT<0000004625A17A68>",
                "Feature":"_invariant",
                "Routine":"",
                "Message":"Routine failure.",
                "Effect":"Fail",
                "Initial_Text":""},
                {"Class":"ACCOUNT<0000004625A17A68>",
                "Feature":"withdraw",
                "Routine":"3",
                "Message":"Routine failure.",
                "Effect":"Fail",
                "Initial_Text":""},
                {"Class":"APPLICATION<0000004625A175A8>",
                "Feature":"make",
                "Routine":"5",
                "Message":"Routine failure.",
                "Effect":"Fail",
                "Initial_Text":""},
                {"Class":"APPLICATION<0000004625A175A8>",
                "Feature":"root's creation",
                "Routine":"",
                "Message":"Routine failure.",
                "Effect":"Exit","Initial_Text":""}]
}]
```


The Runtime_Error has the following keys

- Class,
- Feature,
- Routine(Line number of that feature where the exception occurred) if any,
- Message,
- Effect.

The above is repeated for every stack trace.

In addition to this, all the runtime_error has an extra key initial_text. The initial_text of only the first JSON_OBJECT is useful(it gives the additional runtime messages before the stack, its heading, etc).
(Runtime_Error[0].Initial_Text)

Flat View

http://localhost:9090/flatView?id=”id-given”; class=”your_class”””

When you want the flat view of a project, you must always supply the project id in the URL. The server already has the path information stored with it so it does not need it.

The response object JSON for flat_view looks something like this:


```
 [{
         "Flat_View":"……..",
         "Error_Message":"",
         "Warning_Message":"Warning code: Unused_local_warning\n\nWarning: unreferenced local
variable(s)\nWhat to do: Remove it if you don't plan to use it in the future.\n\nClass:
APPLICATION\nFeature: make\nUnused local is: \n\tc: INTEGER_32\n\n------------------------------
-----------------------------------------------",
         "Syntax Message":"",
         "Dump":"Eiffel Compilation Manager\nVersion 14.05.9.5158 GPL Edition - win64\n\nDegree
6: Examining System\nDegree 5: Parsing Classes\nDegree 4: Analyzing Inheritance\nDegree 3:
Checking Types\nDegree 2: Generating Byte Code\nDegree 1: Generating Metadata\nMelting System
Changes\n--------------------------------------------------------------------------------
\n\nWarning code: Unused_local_warning\n\nWarning: unreferenced local variable(s)\nWhat to do:
Remove it if you don't plan to use it in the future.\n\nClass: APPLICATION\nFeature: make\nUnused
local is: \n\tc: INTEGER_32\n\n--------------------------------------------------------------------
-------------\nSystem Recompiled.\nnote\ Flat_View here",
         "Errors":null,
         "Warnings":[{
                  "Warning_Code":"Unused_local_warning\n",
                  "Warning":"unreferenced local variable(s)\n",
                  "What_to_do":"Remove it if you don't plan to use it in the future.\n\n",
                  "Class":"APPLICATION\n",
                  "Feature":"make\n",
                  "After Feature":"Unused local is: \n\tc: INTEGER 32\n\n",
                  "Dump":"Warning code: Unused_local_warning\n\nWarning: unreferenced local
variable(s)\nWhat to do: Remove it if you don't plan to use it in the future.\n\nClass:
APPLICATION\nFeature: make\nUnused local is: \n\tc: INTEGER_32\n\n"}]}]
```


The flat view has the following keys:

Flat_View: contains the flat view
Error_Message : contains the unparsed compile time error message
Warning_Message: contains the unparsed warning message
Syntax_Message: contains the unparsed syntax message
Dump: Contains the whole dump of the compiler including the flat view, errors, warnings, etc
Errors: Contains the parsed error JSON object with the same keys as that of compilation
Warnings: Contains the parsed warning JSON object with the same keys as that of compilation

Contract View

http://localhost:9090/contractView?id="id-given"; class="your_class"""

When you want the contract view of a project, you must always supply the project id in the URL. The server already has the path information stored with it so it does not need it.

The response object JSON for contract_view looks something like this:


```
[{
        "Contract_View":"note\n\tdescription: \"Summary description for {ACCOUNT}.\"\n\tauthor:
\"\"\n\tdate: \"$Date$\"\n\trevision: \"$Revision$\"\n\nclass interface\n\tACCOUNT\n\ncreate
\n\tmake\n\nfeature \n\n\tbalance: REAL_64\n\n\tdeposit (amt:
REAL_64)\n\t\trequire\n\t\t\tamt_positive: amt > 0.to_double\n\t\tensure\n\t\t\t\tbalance = old
balance + amt\n\n\tmake (a: REAL_64)\n\t\trequire\n\t\t\tbal_positive: a >
0.to_double\n\t\tensure\n\t\t\tbal_set: balance = a\n\n\twithdraw (amt:
REAL_64)\n\t\trequire\n\t\t\tamt positive: amt > 0.to double\n\t\tensure\n\t\t\t\tbalance = old
balance - amt\n\t\ninvariant\n\tbalance_positive: balance > 0.to_double\n\nend -- class
ACCOUNT\n\n",
        "Error_Message":"",
        "Warning Message":"",
        "Syntax_Message":"",
        "Dump":"Eiffel Compilation Manager\nVersion 14.05.9.5158 GPL Edition - win64\n\nDegree
6: Examining System\nSystem Recompiled.\nnote\n\tdescription: \"Summary description for
{ACCOUNT}.\"\n\tauthor: \"\"\n\tdate: \"$Date$\"\n\trevision: \"$Revision$\"\n\nclass
interface\n\tACCOUNT\n\ncreate \n\tmake\n\nfeature \n\n\tbalance: REAL_64\n\n\tdeposit (amt:
REAL 64)\n\t\trequire\n\t\t\tamt positive: amt > 0.to double\n\t\tensure\n\t\t\t\tbalance = old
balance + amt\n\n\tmake (a: REAL_64)\n\t\trequire\n\t\t\tbal_positive: a >
0.to double\n\t\tensure\n\t\t\tbal set: balance = a\n\n\twithdraw (amt:
REAL_64)\n\t\trequire\n\t\t\tamt_positive: amt > 0.to_double\n\t\tensure\n\t\t\t\tbalance = old
balance - amt\n\t\ninvariant\n\tbalance_positive: balance > 0.to_double\n\nend -- class
ACCOUNT\n\n",
        "Errors":null,
        "Warnings":null}]
```


The contract view has the following keys:

Contract_View: contains the contract view
Error_Message : contains the unparsed compile time error message
Warning_Message: contains the unparsed warning message
Syntax_Message: contains the unparsed syntax message
Dump: Contains the whole dump of the compiler including the contract view, errors, warnings, etc
Errors: Contains the parsed error JSON object with the same keys as that of compilation
Warnings: Contains the parsed warning JSON object with the same keys as that of compilation

Class Descendants

http://localhost:9090/classDescendants?id=”id-given”;class=”your_class””

When you want the class_descendants of a project, you must always supply the project id in the URL. The server already has the path information stored with it so it does not need it.

The response object JSON for classDescendants looks something like this:


```
[{
        "Class_Descendants_Dump":"\n\tAPPLICATION\n\n",
        "Error_Message":"",
        "Warning_Message":"",
        "Syntax_Message":"",
        "Dump":"Eiffel Compilation Manager\nVersion 14.05.9.5158 GPL Edition - win64\n\nDegree
6: Examining System\nSystem Recompiled.\n\n\tAPPLICATION\n\n",
        "Errors":null,
        "Warnings":null,
        "Descendants":[{"Deferred":false,"Class_Name":"APPLICATION","Descendants":[]}]
}]
```


The class_descendants has the following keys:

Class_Descendants_Dump: Gives the unparsed dump of class_descendants
Error_Message : contains the unparsed compile time error message
Warning_Message: contains the unparsed warning message
Syntax_Message: contains the unparsed syntax message
Dump: Contains the whole dump of the compiler including the descendants, errors, warnings, etc
Errors: Contains the parsed error JSON object with the same keys as that of compilation
Warnings: Contains the parsed warning JSON object with the same keys as that of compilation
Descendants: Contains the parsed class_descendants with the following keys:

- Class_Name: the name of the class
- Deferred: If it is a deferred class or not (Boolean)
- Descendants: The JSON_ARRAY containing its descendants, and it has the same format, and it continues.

Possible Errors

If the id is not provided with any URL, it might cause error.
If the path is not provided the first time, it might cause error.
If the class is not provided for a view or descendant, it might cause error.

Using the service

Inside the eve_server/server_app/www/js folder there is app.js
All the requests are made inside that app.js for the GET methods. You can change them to suit your needs.

For a new project, provide the PATH in the URL. The new projects are created in eve_server/projects/
Then extract the id given in the request, and send the id without any path the next time.
Right now, I was using a created project with the id that I received.