Documentation
for the
Eiffel Backend Service

Compilation

http://localhost:9090/compile?clean=true;path=your_path;id=your_id;target=your_target

Parameter Name	Optional	Default Value (if optional)	Type
clean	Yes	false	Boolean
path	Not for the first time a project is compiled		String
id	Only for the first time a project is compiled. After that you always have to supply a VALID id		String
target	If the project has multiple targets, then it is non-optional for the first time. If the project has only a single target, then this field is optional.		String

The first time you use a project, you have to provide a non-empty 'path'. The 'id' field may be blank or not present. The clean field may be true/ false or not present. The target field must be non-empty for a multiple-target project, otherwise it can be blank or not present in a single target project.
The next time, you have to provide a VALID id. You may omit 'clean', and the 'path' may be empty or not present. The target may be blank or not present, but if you want to change the target, you can specify it.
The first time a project is compiled, it will always undergo clean compilation.
The server will create a new project directory for you in ./projects/ and return the unique id in the "id" key of the response object.

Targets: Whenever you provide a target, the server stores it and uses it as the default target for all other functions from the API, unless you specify a new target and compile using that new target.
You can only change your default target by calling /compile with the new target. No other calls change the default target.
If the target is invalid, the server responds with the list of available targets. You MUST compile with the right target first, and then use any other functions from the API. If the server has a wrong target stored and you try to use other functions from the API, it will result in an error.

Note: Whenever you provide a path in the arguments, it will always do a clean compilation of the project.

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":"Eiffel C/C++ Compilation Tool - Version: 14.05\nCopyright Eiffel
Software 1996-2012. All Rights Reserved",
	"Error_Message":"",
	"Has_Compilation_Error":false,
	"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-------------------------------
-----------------------------------------------",
	"Has_Warning":true,
	"Needs_Target":false,
	"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\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.\nYou must
	now run \"finish_freezing\"
	in:\n\tC:\\Users\\Manav\\Desktop\\eve_server\\projects\\sample_90799D1A-C197-4822-BEE9-
	C0D076CA8562\\EIFGENs\\sample\\W_code\n",
	"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"
		}],
	"Targets":null,
	"Compilation_Succeeded":true
}]
```


The keys in the JSON response for compilation

Name of Key	Type	Description	Default Values
Compile_Message	String	Gives the compile message	“”
Output_Message	String	Gives the C compilation results and any other output messages if any	“”
Error_Message	String	Gives the unparsed error message	“”
Has_Compilation_Error	Boolean	Indicates whether there is an error or not in the compilation	True/false
Warning_Message	String	Gives the unparsed warning message	“”
Has_Warning	Boolean	Indicates whether there is a warning message in the compilation.	True/false
Needs_Target	Boolean	Indicates that the user either specified a	True/False

		wrong target or did not specify a target for a multi-target project. This field MUST always be false before you use another function from the API, otherwise it will give an error	
Dump	String	Gives the entire dump message with errors, warnings, and the compile results.	""
Errors	Json Array	Gives the parsed JSON object for the error message with its keys	Null
Warnings	Json Array	Gives the parsed JSON object for the warning message with its keys	Null
Targets	Json Array	Gives the parsed JSON object for the list of available targets for the project	Null
Compilation_Succeeded	Boolean	Indicates whether the compilation was successful or not. The compilation fails under the following circumstances: 1. Compilation Error : You can check the Has_Compilation_Error field, which will be true. 2. Wrong/ Missing Target: You can check the Needs_Target field, which will be true. 3. Timeout: You can check the status code for timeouts (504).	True/False

The keys for the Warnings in the JSON response for compilation

Name of Key	Type	Description	Default Value(if not present in the warning message)
Warning_Code	String	Gives the warning code	""
Warning	String	Gives the warning message	""
What_to_do	String	Gives the what to do string	""
Class	String	Gives the name of the class where the warning occured	""
Feature	String	Gives the name of the feature where the warning occured	""
After_Feature	String	Gives the part after the feature in the warning message	""
Dump	String	Gives the dump of this warning message	Has to be present

The keys for the Error in the JSON response for compilation

Name of Key	Type	Description	Default Value(if not present in the error message)

Error_Code	String	Gives the error code	“”
Error	String	Gives the error message	“”
What_to_do	String	Gives the what to do string	“”
Class	String	Gives the name of the class where the error occured	“”
Feature	String	Gives the name of the feature where the error occured	“”
Line	Integer	Gives the line number where the error occured	-1
Before_Line	String	Gives the part before the line in the error message	“”
After_Line	String	Gives the part after the line in the error message	“”
Dump	String	Gives the dump of this error message	Has to be present

The keys for the Targets in the JSON response for compilation

Name of Key	Type	Description	Default Value(if not present in the error message)
Target	String	Gives the target	“”

Target JSON_array

"Targets":[{"Target":"sampleA"},{"Target":"sampleB"}]

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: APPLICATION\nFeature: extra_feature\nIdentifier: asdas\nTarget
type: [like Current] attached APPLICATION\nLine: 40\n        create s.make_from_string
(\"abcd\")\n->      asdas\n      ensure\n\n------------------------------------------------------
-------------------------",
        "Has_Compilation_Error":true,
        "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------------------------------
-----------------------------------------------",
        "Has_Warning":true,
        "Needs_Target":false,
        "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\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------------------------------
-----------------------------------------------\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: APPLICATION\nFeature:
extra_feature\nIdentifier: asdas\nTarget type: [like Current] attached APPLICATION\nLine: 40\n
```

```
create s.make_from_string (\"abcd\")\n->      asdas\n      ensure\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":"APPLICATION\n",
                        "Feature":"extra_feature\n",
                        "Line":40,
                        "Before_Line":"Identifier: asdas\nTarget type: [like Current] attached
                APPLICATION\n",
                        "After_Line":"        create s.make_from_string (\"abcd\")\n->
                asdas\n      ensure\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: APPLICATION\nFeature:
                extra_feature\nIdentifier: asdas\nTarget type: [like Current] attached
                APPLICATION\nLine: 40\n        create s.make_from_string (\"abcd\")\n->
                asdas\n      ensure\n\n"}],
        "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"
                }]}],
        "Targets":null,
        "Compilation_Succeeded":false
}]
```


Note 1 : After the server is started, it needs to set the 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 gives "You don't have the permission to write to its EIFGENS's" error)

Execution

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

Parameter Name	Optional	Default Value (if optional)	Type
id	No		String

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 JSON response object for error free execution looks like this:

```
[{
        "Execution_Output":"Hello Eiffel World!\nNumber not out of range\n0",
        "Error_Message":"",
        "Warning_Message":"",
        "Has_Warning":false,
        "Compile_Errors":null,
        "Has_Compilation_Error":false,
        "Warnings”:null,
        “Runtime_Text”:””,
        "Runtime_Errors":null,
        "Has_Runtime_Error":false,
        "Execution_Succeeded":true,
}]
```

The keys in the JSON response for execution

Name of Key	Type	Description	Default Value
Execution_Output	String	Gives the execution output	“”
Error_Message	String	Gives the unparsed error message (runtime error message if there is a runtime error or compile time error message if there is a compile time error)	“”
Warning_Message	String	Gives the unparsed warning message	“”
Has_Warning	Boolean	Indicates whether there is a warning message in the compilation.	True/false
Compile_Errors	Json Array	Gives the parsed JSON object for the compile time error message with its keys	Null
Has_Compilation_Error	Boolean	Indicates whether there is a compilation error or not	True/false
Warnings	Json Array	Gives the parsed JSON object for the warning message with its keys	Null
Runtime_Text	String	Gives the heading and message at the top of the table of the stack trace	“”
Runtime_Errors	Json Array	Gives the parsed JSON object for the runtime error message with its keys	Null
Has_Runtime_Error	Boolean	Indicates whether there is a runtime error or	True/false

		not	
Execution_Succeeded	Boolean	Indicates whether the execution was successful or not. The execution fails under the following circumstances: 1. Compilation Error : You can check the Has_Compilation_Error field, which will be true. 2. Runtime Error: You can check the Has_Runtime_Error field, which will be true. 3. Timeout: You can check the status code for timeouts (504).	

The keys for the Runtime_Errors in the JSON response for execution

Name of Key	Type	Description
Class	String	Gives the class name
Feature	String	Gives the feature name
Routine	String	Gives the position in the feature of the class where the runtime exception occured
Message	String	Gives the message of the runtime error
Effect	String	Gives the effect of the runtime exception

The above JSON-OBJECT is repeated for every stack trace, thereby making the Runtime_Error JSON_ARRAY.

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
@2          balance_positive:           \n<00000056A6366A28>                        Class
invariant violated.     Fail\n-------------------------------------------------------------------------
------------\nACCOUNT             _invariant
\n<00000056A6366A28>                        Routine failure.              Fail\n----------------
---------------------------------------------------------------\nACCOUNT             withdraw @3
\n<00000056A6366A28>                        Routine failure.              Fail\n----------------
---------------------------------------------------------------\nAPPLICATION         make @5
\n<00000056A6366588>                        Routine failure.              Fail\n----------------
---------------------------------------------------------------\nAPPLICATION         root's
creation                                    \n<00000056A6366588>                         Routine
failure.              Exit\n-------------------------------------------------------------------------
----------\n",
        "Warning_Message":"",
        "Has_Warning":false,
        "Compile_Errors":null,
        "Has_Compilation_Error":false,
```

```
"Warnings":null,
"Runtime_Text":"\nsample: system execution failed.\nFollowing is the set of recorded
exceptions:\n",
"Runtime_Errors":[{
        "Class":"ACCOUNT<00000056A6366A28>",
        "Feature":"_invariant",
        "Routine":"2",
        "Message":"balance_positive:Class invariant violated.",
        "Effect":"Fail"},
        {"Class":"ACCOUNT<00000056A6366A28>",
        "Feature":"_invariant",
        "Routine":"",
        "Message":"Routine failure.",
        "Effect":"Fail”},
        {"Class":"ACCOUNT<00000056A6366A28>",
        "Feature":"withdraw",
        "Routine":"3",
        "Message":"Routine failure.",
        "Effect":"Fail"},
        {"Class":"APPLICATION<00000056A6366588>",
        "Feature":"make",
        "Routine":"5",
        "Message":"Routine failure.",
        "Effect":"Fail”},
        {"Class":"APPLICATION<00000056A6366588>",
        "Feature":"root's creation",
        "Routine":"",
        "Message":"Routine failure.",
        "Effect":"Exit"
}],
"Has_Runtime_Error":true,
"Execution_Succeeded":false }]
```

Flat View

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

Parameter Name	Optional	Default Value (if optional)	Type
id	No		String
class	No		String

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

The JSON response object JSON for flat_view looks like this:


```
 [{
         "Flat_View":"…….",
         "Error_Message":"",
         "Has_Compilation_Error":false,
         "Has Flat View":true,
         "Warning_Message":"",
         "Has_Warning":true,
         "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 keys in the JSON response for flat view

Name of Key	Type	Description	Default Values
Flat_View	String	Gives the Flat View of the class	""
Error_Message	String	Gives the unparsed error message	""
Has_Compilation_Error	Boolean	Indicates whether there is an error or not in the compilation	True/false
Has_Flat_View	Boolean	Indicates whether the response object contains the flat view or not. The flat view is not shown under the following circumstances: 1. Compilation Error : You can check	True/False

		the Has_Compilation_Error field, which will be true. 2. Class does not exist: In this case Has_Compilation_Error is false, but the error message is non-empty and contains the error message for the class does not exist 3. Timeout: You can check the status code for timeouts (504)	
Warning_Message	String	Gives the unparsed warning message	“”
Has_Warning	Boolean	Indicates whether there is a warning message in the compilation.	True/false
Dump	String	Gives the entire dump message with errors, warnings, and the flat view.	“”
Errors	Json Array	Gives the parsed JSON object for the error message with its keys	Null
Warnings	Json Array	Gives the parsed JSON object for the warning message with its keys	Null

Contract View

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

Parameter Name	Optional	Default Value (if optional)	Type
id	No		String
class	No		String

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

The JSON response object JSON for contract_view looks 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":"",
        "Has_Compilation_Error":false,
        "Has_Contract_View":true,
        "Warning_Message":"",
        "Has_Warning":false,
        "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 keys in the JSON response for flat view

Name of Key	Type	Description	Default Values
Contract_View	String	Gives the Contract View of the class	""
Error_Message	String	Gives the unparsed error message	""
Has_Compilation_Error	Boolean	Indicates whether there is an error or not in the compilation	True/false
Has_Contract_View	Boolean	Indicates whether the response object contains the contract view or not. The contract view is not shown under the following circumstances:	True/False

		1. Compilation Error : You can check the Has_Compilation_Error field, which will be true. 2. Class does not exist: In this case Has_Compilation_Error is false, but the error message is non-empty and contains the error message for the class does not exist 3. Timeout: You can check the status code for timeouts (504)	
`Warning_Message`	`String`	Gives the unparsed warning message	“”
`Has_Warning`	`Boolean`	Indicates whether there is a warning message in the compilation.	True/false
`Dump`	`String`	Gives the entire dump message with errors, warnings, and the flat view.	“”
`Errors`	`Json Array`	Gives the parsed JSON object for the error message with its keys	Null
`Warnings`	`Json Array`	Gives the parsed JSON object for the warning message with its keys	Null

Class Descendants

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

Parameter Name	Optional	Default Value (if optional)	Type
id	No		String
class	No		String

When you want the descendants of a class, you must always supply the 'id' and the 'class' 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 like this:


```
[{
        "Class_Descendants_Dump":"\n\tAPPLICATION\n\n",
        "Error_Message":"",
        "Has_Compilation_Error":false,
        "Has_Descendants":true,
        "Warning_Message":"",
        "Has_Warning":false,
        "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":[{"Deferred":f
        alse,"Class_Name":"Class_A","Descendants":[]},{"Deferred":false,"Class_Name":"Class_B","
        Descendants":[]}]
        }]
}]
```


The keys in the JSON response for class descendants

Name of Key	Type	Description	Default Values
Class_Descendants_Dump	String	Gives the dump of the Descendants of the class	""
Error_Message	String	Gives the unparsed error message	""
Has_Compilation_Error	Boolean	Indicates whether there is an error or not in the compilation	True/false
Has_Descendants	Boolean	Indicates whether the response object contains the descendants or not The descendants is not shown under the following circumstances: 1. Compilation Error : You can check the Has_Compilation_Error field, which will be true. 2. Class does not exist: In this case Has_Compilation_Error is false, but the error message is non-empty and contains the error message for	True/False

		the class does not exist 3. Timeout: You can check the status code for timeouts (504)	
Warning_Message	String	Gives the unparsed warning message	“”
Has_Warning	Boolean	Indicates whether there is a warning message in the compilation.	True/false
Dump	String	Gives the entire dump message with errors, warnings, and the flat view.	“”
Errors	Json Array	Gives the parsed JSON object for the error message with its keys	Null
Warnings	Json Array	Gives the parsed JSON object for the warning message with its keys	Null
Descendants	Json Array	Gives the JSON array for the parsed descendants structure	Null

The keys in the Descendants JSON array

Name of Key	Type	Description	Default Values
Class_Name	String	Gives the Class Name	“”
Deferred	Boolean	Indicates whether the class is a deferred class or not	True/False
Descendants	Json Array	Gives the descendants of this class	Null

Class Ancestors

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

Parameter Name	Optional	Default Value (if optional)	Type
id	No		String
class	No		String

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

The response object JSON for class Ancestors looks like this:


```
[{
        "Class_Ancestors_Dump":"\n\tAPPLICATION\n\n",
        "Error_Message":"",
        "Has_Compilation_Error":false,
        "Has_Ancestors":true,
        "Warning_Message":"",
        "Has_Warning":false,
        "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,
        "Ancestors":[{"Deferred":false,"Class_Name":"APPLICATION","Ancestors":[{"Deferred":false
        ,"Class_Name":"Class_A","Ancestors":[]},{"Deferred":false,"Class_Name":"Class_B","Ancest
        ors":[]}]
        }]
}]
```


The keys in the JSON response for class ancestors

Name of Key	Type	Description	Default Values
Class_Ancestors_Dump	String	Gives the dump of the Ancestors of the class	""
Error_Message	String	Gives the unparsed error message	""
Has_Compilation_Error	Boolean	Indicates whether there is an error or not in the compilation	True/false
Has_Ancestors	Boolean	Indicates whether the response object contains the ancestors or not. The ancestors is not shown under the following circumstances: 1. Compilation Error : You can check the Has_Compilation_Error field, which will be true. 2. Class does not exist: In this case Has_Compilation_Error is false, but the error message is non-empty and contains the error message for	True/False

		the class does not exist 3. Timeout: You can check the status code for timeouts (504)	
Warning_Message	String	Gives the unparsed warning message	“”
Has_Warning	Boolean	Indicates whether there is a warning message in the compilation.	True/false
Dump	String	Gives the entire dump message with errors, warnings, and the flat view.	“”
Errors	Json Array	Gives the parsed JSON object for the error message with its keys	Null
Warnings	Json Array	Gives the parsed JSON object for the warning message with its keys	Null
Ancestors	Json Array	Gives the JSON array for the parsed ancestors structure	Null

The keys in the Ancestors JSON array

Name of Key	Type	Description	Default Values
Class_Name	String	Gives the Class Name	“”
Deferred	Boolean	Indicates whether the class is a deferred class or not	True/False
Ancestors	Json Array	Gives the ancestors of this class	Null

Class Clients

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

Parameter Name	Optional	Default Value (if optional)	Type
id	No		String
class	No		String

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

The response object JSON for class clients looks like this:


```
[{
        "Class_Clients_Dump":"\n\tAPPLICATION\n\n",
        "Error_Message":"",
        "Has_Compilation_Error":false,
        "Has_Clients":true,
        "Warning_Message":"",
        "Has_Warning":false,
        "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,
        "Clients":[{"Deferred":false,"Class_Name":"SampleA},{"Deferred":false,"Class_Name":"Samp
        leB"}]
}]
```


The keys in the JSON response for class clients

Name of Key	Type	Description	Default Values
Class_Clients_Dump	String	Gives the dump of the clients of the class	""
Error_Message	String	Gives the unparsed error message	""
Has_Compilation_Error	Boolean	Indicates whether there is an error or not in the compilation	True/false
Has_Clients	Boolean	Indicates whether the response object contains the clients or not. The clients is not shown under the following circumstances: 1. Compilation Error : You can check the Has_Compilation_Error field, which will be true. 2. Class does not exist: In this case Has_Compilation_Error is false, but the error message is non-empty and contains the error message for the class does not exist 3. Timeout: You can check the status code for timeouts (504)	True/False

Warning_Message	String	Gives the unparsed warning message	“”
Has_Warning	Boolean	Indicates whether there is a warning message in the compilation.	True/false
Dump	String	Gives the entire dump message with errors, warnings, and the flat view.	“”
Errors	Json Array	Gives the parsed JSON object for the error message with its keys	Null
Warnings	Json Array	Gives the parsed JSON object for the warning message with its keys	Null
Clients	Json Array	Gives the JSON array for the parsed clients structure	Null

The keys in the Clients JSON array

Name of Key	Type	Description	Default Values
Class_Name	String	Gives the Class Name	“”
Deferred	Boolean	Indicates whether the class is a deferred class or not	True/False

Class Suppliers

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

Parameter Name	Optional	Default Value (if optional)	Type
id	No		String
class	No		String

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

The response object JSON for class supplier looks like this:


```
[{
        "Class_Suppliers_Dump":"\n\tAPPLICATION\n\n",
        "Error_Message":"",
        "Has_Compilation_Error":false,
        "Has_Suppliers":true,
        "Warning_Message":"",
        "Has_Warning":false,
        "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,
        "Supliers":[{"Deferred":false,"Class_Name":"SampleA},{"Deferred":false,"Class_Name":"Sam
        pleB"}]
}]
```


The keys in the JSON response for class clients

Name of Key	Type	Description	Default Values
Class_Suppliers_Dump	String	Gives the dump of the suppliers of the class	""
Error_Message	String	Gives the unparsed error message	""
Has_Compilation_Error	Boolean	Indicates whether there is an error or not in the compilation	True/false
Has_Suppliers	Boolean	Indicates whether the response object contains the suppliers or not. The suppliers is not shown under the following circumstances: 1. Compilation Error : You can check the Has_Compilation_Error field, which will be true. 2. Class does not exist: In this case Has_Compilation_Error is false, but the error message is non-empty and contains the error message for the class does not exist 3. Timeout: You can check the status code for timeouts (504)	True/False

Warning_Message	String	Gives the unparsed warning message	“”
Has_Warning	Boolean	Indicates whether there is a warning message in the compilation.	True/false
Dump	String	Gives the entire dump message with errors, warnings, and the flat view.	“”
Errors	Json Array	Gives the parsed JSON object for the error message with its keys	Null
Warnings	Json Array	Gives the parsed JSON object for the warning message with its keys	Null
Suppliers	Json Array	Gives the JSON array for the parsed suppliers structure	Null

The keys in the Suppliers JSON array

Name of Key	Type	Description	Default Values
Class_Name	String	Gives the Class Name	“”
Deferred	Boolean	Indicates whether the class is a deferred class or not	True/False

Feature Callers

http://localhost:9090/featureCallers?id=id-given;class=your_class;feature=your_feature

Parameter Name	Optional	Default Value (if optional)	Type
id	No		String
class	No		String
Feature	No		String

When you want the feature callers of a feature of a class, you must always supply the 'id', 'class' and 'feature' in the URL. The server already has the path information stored with it so it does not need it.

The response object JSON for feature caller looks like this:


```
[{
        "Feature_Callers_Dump":"balance from
ACCOUNT\n\tACCOUNT\n\t\tdeposit\n\t\tinvariant\n\t\tmake\n\t\twithdraw\n\n",
        "Error_Message":"",
        "Has_Compilation_Error":false,
        "Has_Feature_Callers":true,
        "Warning_Message":"",
        "Has_Warning":false,
        "Dump":"Eiffel Compilation Manager\nVersion 14.05.9.5158 GPL Edition - win64\n\nDegree
6: Examining System\nSystem Recompiled.\nbalance from
ACCOUNT\n\tACCOUNT\n\t\tdeposit\n\t\tinvariant\n\t\tmake\n\t\twithdraw\n\n",
        "Errors":null,
        "Warnings":null,
        "Callers":[{"Class_Name":"ACCOUNT","Features":[{"Feature_Name":"deposit"},{"Feature_Name
":"invariant"},{"Feature_Name":"make"},{"Feature_Name":"withdraw"}]}]
}]
```


The keys in the JSON response for class clients

Name of Key	Type	Description	Default Values
Feature_Callers_Dump	String	Gives the dump of the callers of the feature of the class	""
Error_Message	String	Gives the unparsed error message	""
Has_Compilation_Error	Boolean	Indicates whether there is an error or not in the compilation	True/false
Has_Feature_Callers	Boolean	Indicates whether the response object contains the feature callers or not. The feature callers is not shown under the following circumstances: 1. Compilation Error : You can check the Has_Compilation_Error field, which will be true. 2. Class/Feature does not exist: In this case Has_Compilation_Error is false, but the error message is non-empty and contains the error	True/False

		message for the class does not exist 3. Timeout: You can check the status code for timeouts (504)	
Warning_Message	String	Gives the unparsed warning message	“”
Has_Warning	Boolean	Indicates whether there is a warning message in the compilation.	True/false
Dump	String	Gives the entire dump message with errors, warnings, and the flat view.	“”
Errors	Json Array	Gives the parsed JSON object for the error message with its keys	Null
Warnings	Json Array	Gives the parsed JSON object for the warning message with its keys	Null
Callers	Json Array	Gives the JSON array for the parsed callers structure	Null

The keys in the Callers JSON array

Name of Key	Type	Description	Default Values
Class_Name	String	Gives the Class Name	“”
Features	Json Array	Gives the list of features of Class_Name that call the requested feature	Null

The keys in the Features JSON array

Name of Key	Type	Description	Default Values
Feature_Name	String	Gives the Feature Name	“”

Command Line

http://localhost:9090/commandLine?id=id-given;command_line=your_message

Parameter Name	Optional	Default Value (if optional)	Type
id	No		String
command_line	No		String

When you want to provide a command line argument, you must always supply the 'id'and the 'command_line' in the URL. The server already has the path information stored with it so it does not need it.

The response object JSON for command line output for the command_line="ec –config sample.ecf –callers ACCOUNT balance" looks like this:


```
[{
        "Command_Line_Dump":"balance from
ACCOUNT\n\tACCOUNT\n\t\tdeposit\n\t\tinvariant\n\t\tmake\n\t\twithdraw\n\n
}]
```


The keys in the JSON response for class clients

Name of Key	Type	Description	Default Values
Command_Line_Dump	String	Gives the dump of the output from the compiler after executing the command line argument given by the user	""

Compilation and Runtime timeout

The server would respond with a 504 Gateway Error and the normal response body, if the compilation or execution time exceeded the values set for the respective timeouts.
Inside ./www, is a file config.json
Change the Compilation_Timeout and Runtime_Timeout values (in seconds) for your need.

Currently the keys and their values are

Key	Value (in seconds)	Type
Compilation_Timeout	180	Integer
Runtime_Timeout	10	Integer

An Example of execution timeout response from the server is:


```
{
	"data":[{
		"Execution_Output":"Hello Eiffel World!\r\nNumber not out of
	range\r\n000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
	00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
	00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
	000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",
		"Error_Message":"",
		"Warning_Message":"",
		"Has_Warning":false,
		"Compile_Errors":null,
		"Has_Compilation_Error":false,
		"Warnings":null,
		"Runtime_Text":"",
		"Runtime_Errors":null,
		"Has_Runtime_Error":false,
		"Execution_Succeeded":false}],
	"status":504,
	"config":{
		"method":"GET",
		"transformRequest":[null],
		"transformResponse":[null],
		"url":"/run",
		"params":{"id":"90799D1A-C197-4822-BEE9-C0D076CA8562"},
		"headers":{"Accept":"application/json, text/plain, */*"}
		}
}
```


The server returns the above response when a 504 is occurred.

The keys in the JSON response for 504 from the server is

Name of Key	Type	Description	Default Values
data	Json Array	Contains the JSON array for the corresponding function (eg, execution in this case). The Execution_Output(in runtime_timeout) and Compile_Message(in compilation_timeout) are displayed till the point they were not timed out.	null
status	Integer	Gives the status Code	504 (always in our case)
config	Json Object	Gives the method, url, transformRequest, transformResponse, params, and headers.	

Response Status Codes

Status Code	Description
200	Ok
504	Gateway Timeout – The server did not get a response from the upstream server on time. Represents a compilation or a runtime timeout.

Removing Old EIFGEN's

The server creates a new thread that runs continuously. It checks once daily (fixed time, say midnight) the user has not used a particular project for a fixed long time (say 90 days). If it is true, it deletes the project's EIFGEN's to save memory.
The developer can set these fixed values by changing them in the config.json file inside ./www folder.

Currently the keys and their values are

Key	Value (in seconds)	Type	Description
Sleep_Time	5000000 (in nanoseconds)	Integer	It defines the time (in seconds) that the thread checking for old EIFGEN's should sleep, if a conflict occurs between it and the main thread which responds with user requests.
EIFGEN_Duration	7776000 (90 days)	Integer	It defines the time after which a user project is considered inactive.
Seconds_After_Midnight	10	Integer	It defines the time (in seconds) after midnight, when the server checks for old projects.

If suppose the side thread (checking for old projects) and the main thread are simultaneously working on the same project, the main thread suspends the side_thread before calling the command line for Sleep_Time seconds, thus avoiding a clash.

Possible Errors

- If the id is not provided with any URL, it will cause the program to crash.
- If the path is not provided the first time, it will cause the program to crash.
- If the class is not provided for any class function which requires a class, it will cause the program to crash.
- If the feature is not provided for any feature function which requires a feature, it will cause the program to crash.
- If any function from the API (except) compile/clean compile is called with a wrong_target name/ no target_name for a multi-target project, it will produce an error.