SWMM-Docs

5.2.0.dev4

Generated by Doxygen 1.8.13

Contents

Chapter 1

Open Water Analytics Stormwater Management Model

The OWA-SWMM Open-Source Library is a hydraulic, hydrologic, and water quality analysis model, originally developed by USEPA, written in C.

If you are interested in extending OWA-SWMM for academic, personal, or commercial use, then you've come to the right place. For community discussion and contributions you might have that could benefit a larger base of developers and users, please raise some issues over at `SWMM Issues`

We welcome discussion of all kinds! The OWA community of developers is excited to work with you! SERIOUSLY!!! If you like coding, join in on the discussions and share some ideas and code!

The OWA-SWMM project builds on top of the contributions put forward by Lew Rossman with his original work on delivering EPA-SWMM5. We are grateful for all the work that has been put into SWMM5 and we are committed to delivering advances to the engine in a community driven approach.

- Toolkit Overview
- MIT-License
- Authors at Open Water Analytics

Chapter 2

Toolkit Overview

The Programmer's Toolkit application programming interface (API) is an extension of the SWMM simulation package. Through the API, a developer can programmatically talk to SWMM before, during, and after a simulation. The API exposes SWMM's data model which enables new possibility. When a hydraulic network is initialized, a network is most generally comprised of nodes, link, and runoff surfaces. Nodes and Links are normally manholes and conduits, and runoff surfaces are a hydrologic representation of rainfall catchment surfaces that send flow into the hydraulic network.

The Toolkit API provides a series of functions that allow programmers to customize the use of SWMM's hydraulic and water quality solution engine to their own applications. Before using the Toolkit one should become familiar with the way that SWMM represents the hydrologic surfaces and hydraulic network and the design and operating information it requires to perform a simulation. This information can be obtained from reading SWMM's Users Manuals `USEPA Link`.

A typical usage of the Toolkit functions to analyze a stormwater network might look as follows:

1. Use the swmm_open function to open the Toolkit system, along with a SWMM [Input file](Input-File).
2. Use the `swmm_setxxxParam` to update parameters before beginning a simulation
3. Use the swmm_start function to start the simulation.
4. Iterator over the simlation using swmm_step function and exit the loop when a 0 is returned. While the simlation is running, use the `swmm_getxxxResult` or swmm_getxxxStats‘ to read results.
5. Use the swmm_end function to end a simulation and save the results.
6. Use the swmm_close function to close the simulation and free the memory allocated from the simulation.

- How to Use the Toolkit

2.1 How to Use the Toolkit

Full Function collection can be found toolkitAPI.h

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include "swmm5.h"
#include "toolkitAPI.h"

long newHour, oldHour = 0;
long theDay, theHour;
double elapsedTime = 0.0;

char *inputFile;
char *reportFile;
char *binaryFile;

inputFile = "<path2>/inputfile.inp";
reportFile = "<path2>/reportfile.rpt";
binaryFile = "<path2>/outputfile.out";

// Open the files & read input data
ErrorCode = swmm_open(inputFile, reportFile, binaryFile);

// Run the simulation if input data OK
if ( !ErrorCode )
{
    int ndType;
    double depth = 0;

    swmm_getNodeType(0, &ndType);

    printf("Node Type: %d", ndType); \\ Print node type (See SM_NodeType)

    // Initialize values and Start the Simulation
    ErrorCode = swmm_start(TRUE);

    // Execute each time step until elapsed time is re-set to 0
    if ( !ErrorCode )
    {
        do
        {
            ErrorCode = swmm_step(&elapsedTime);

            swmm_getNodeResult(0, SM_NODEDEPTH, &depth) \\ Stream Results!

            printf("Node Depth %lf", depth); \\ Print node result (See
      SM_NodeResult)

        } while ( elapsedTime > 0.0 && !ErrorCode );
    }
    // Clean up
    ErrorCode = swmm_end();
}

// Get Stats for Node
SM_NodeStats* Node0Stats;
swmm_getNodeStats(0, &Node0Stats);

printf("Node Max Depth %lf", Node0Stats->avgDepth);

// Report results
swmm_report();

// Close the system
swmm_close();
```

Chapter 3

Contributor Covenant Code of Conduct

Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, gender identity and expression, level of experience, nationality, personal appearance, race, religion, or sexual identity and orientation.

Our Standards

Examples of behavior that contributes to creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery and unwelcome sexual attention or advances
- Trolling, insulting/derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or electronic address, without explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

Our Responsibilities

Project maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Project maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project team at `bemcdonnell@gmail.com`. The project team will review and investigate all complaints, and will respond in a way that it deems appropriate to the circumstances. The project team is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

Project maintainers who do not follow or enforce the Code of Conduct in good faith may face temporary or permanent repercussions as determined by other members of the project's leadership.

Attribution

This Code of Conduct is adapted from the `Contributor Covenant`, version 1.4, available at `http↩ ://contributor-covenant.org/version/1/4`

Chapter 4

Building OWA SWMM From Source on Windows

by Michael E. Tryby

Created on: March 13, 2018

Introduction

Building OWA's fork of SWMM from source is a basic skill that all developers interested in contributing to the project should know how to perform. This document describes the build process step-by-step. You will learn

1. how to configure your machine to build the project locally;
2. how to obtain the project files using git;
3. how to use cmake to generate build files and build the project; and
4. how to use ctest and nrtest to perform unit and regression testing on the build artifacts produced.

Be advised, you will need local admin privileges on your machine to follow this tutorial. Let's begin!

Dependencies

Before the project can be built the required tools must be installed. The OWA SWMM project adheres to a platform compiler policy - for each platform there is a designated compiler. The platform compiler for Windows is Visual Studio cl, for Linux gcc, and for Mac OS clang. These instructions describe how to build SWMM on Windows. CMake is a cross platform build, testing, and packaging tool that is used to automate the SWMM build workflow. Boost is a free portable peer-reviewed C++ library. Unit tests are linked with Boost unit test libraries. Lastly, git is a free and open source distributed version control system. Git must be installed to obtain the project source code from the OWA SWMM repository found on GitHub.

Summary of Dependencies

- Platform Compiler
  - Windows: Visual Studio 10.0 32-bit cl (version 16.00.40219.01 for 80x86)
- CMake (version 3.0.0 or greater)
- Boost Libraries (version 1.58 or greater)
- git (version 2.6.0 or greater)

Step 1 - Install Dependencies

Install Visual Studio 2010 Express and SP1

Our current benchmark platform and compiler is Windows 32-bit Visual Studio 10 2010. Older versions of Visual Studio are available for download here:

https://www.visualstudio.com/vs/older-downloads/

A service pack for Visual Studio 10 2010 is available here:

https://www.microsoft.com/en-us/download/details.aspx?id=34677

Install Boost

Boost binaries for Windows offer a convenient installation solution. Be sure to select download the boost installer exe that corresponds to the version of Visual Studio you have installed.

https://sourceforge.net/projects/boost/files/boost-binaries/1.58.0/

Although newer version of Boost are available, a link to Boost 1.58 is provided. This is the library version that the unit tests have been written against. Older versions of Boost may not work. The default install location for the Boost

Libraries is `C:\local\boost_1_58_0`

Install Chocolatey, CMake, and git

Chocolatey is a Windows package manager that makes installing some of these dependencies a little easier. When working with Chocolatey it is useful to have local admin privileges. Chocolatey is available here:

https://chocolatey.org/install

Once Chocolately is installed, from a command prompt running with admin privileges issue the following commands

```
\>choco install -y cmake --installargs 'ADD_CMAKE_TO_PATH=User'
\>choco install -y git --installargs /GitOnlyOnPath
\>refreshenv
```

Common Problems

Using chocolatey requires a command prompt with admin privileges. Check to make sure installed applications are on the command path. Make note of the Boost Library install location.

Step 2 - Build The Project

As administrator open a Visual Studio 2010 Command Prompt. Change directories to the location where you wish to build the SWMM project. Now we will issue a series of commands to create a parent directory for the project root and clone the project from OWA's GitHub repository.

Clone the SWMM Repository

```
\>mkdir OWA
\>cd OWA
\>git clone https://github.com/OpenWaterAnalytics/Stormwater-Management-Model.git
\>cd Stormwater-Management-Model
```

The present working directory is now the project root SWMM. The directory contains the same files that are visibly present in the GitHub Repo by browsing to the URL https://github.com/OpenWater↩Analytics/stormwater-management-model/tree/develop.

Now we will create a build products directory and generate the platform build file using cmake.

Generate the build files

```
\>mkdir buildprod
\>cd buildprod
\>set BOOST_ROOT=C:\local\boost_1_58_0
\>cmake -G "Visual Studio 10 2010" -DBOOST_ROOT="%BOOST_ROOT%" -DBoost_USE_STATIC_LIBS="ON" ..
```

Now that the dependencies have been installed and the build system has been generated, building SWMM is a simple CMake command.

Build SWMM

```
\>cmake --build . --config Debug
```

Common Problems

CMake may not be able to find the project CMakeLists.txt file or the Boost library install location.

Step 3 - Testing

Unit Testing uses Boost Unit Test library and CMake ctest as the test runner. Cmake has been configured to register tests with ctest as part of the build process.

Unit Testing

```
\>cd tests
\>ctest -C Debug
```

The unit tests run quietly. Ctest redirects stdout to a log file which can be found in the "tests\Testing\Temporary" folder. This is useful when a test fails.

Regression testing is somewhat more complicated because it relies on Python to execute SWMM for each test and compare the binary files and report files. To run regression tests first python and any required packages must be installed. If Python is already installed on your local machine the installation of miniconda can be skipped.

Installing Regression Testing Dependencies

```
cd ..\..
\>choco install -y miniconda --installargs '/AddToPath=1'
\>refreshenv
\>pip install -r tools/requirements-appveyor.txt
```

With Python and the necessary dependencies installed, regression testing can be run using the gen-config and run-nrtest helper scripts found in the tools folder. The script gen-config creates a json formatted file that describes the build artifact under test - run-swmm - and how to run it. The script run-nrtest calls nrtest execute and nrtest compare to perform the regression test.

To run the executable under test, nrtest needs the absolute path to it and a unique identifier for it such as the version number. The project build places build artifacts in the `buildprod\bin\` folder. On Windows the build configuration "Debug" or "Release" must also be indicated. On Windows it is also necessary to specify the path to the Python Scripts folder so the nrtest execute and compare commands can be found. You need to substitute bracketed fields below like "<build id>" with the values for your setup.

Regression Testing

```
\>tools/gen-config.cmd <Absolute path to exe under test> > tests/apps/swmm-<build id>.json
\>tools/run-nrtest.cmd <Absolute path to python scripts> /tests/swmm-nrtestsuite <build id>
```

Common Problems

The nrtest script complains that it can't find manifest files.

That concludes this tutorial on building OWA SWMM from source on Windows. You have learned how to configure your machine satisfying project dependencies and how to acquire, build, and test SWMM on your local machine. To be sure, there is a lot more to learn, but this is a good start! Learn more by following the links provided below.

Further Reading

Visual Studio - https://msdn.microsoft.com/en-us/library/dd831853(v=vs.100).aspx CMake - https://cmake.org/documentation/ Boost - http://www.boost.org/doc/ git - https://git-scm.com/doc Miniconda - https://conda.io/docs/user-guide/index.html nrtest - https://nrtest.readthedocs.io/en/latest/

Chapter 5

SWMM Regression Testing

Prerequisits

Running SWMM's regression test suite `swmm-nrtestsuite` requires installation of the following software.

- git
- C compiler - MSVC, gcc, xcode
- cmake
- python 2.7 including setup tools
- swig

Step by Step Guide for Linux and MacOS

The following are step by step instructions to compare the current SWMM OWA build against the SWMM 5.1.12 MSVC 32 bit benchmark.

1. Clone the swmm github repository.

   ```
   $ git clone --branch=develop https://github.com/OpenWaterAnalytics/Stormwater-Management-Model.git
   ```

2. Make repository root the current working directory

   ```
   $ cd Stormwater-Management-Model
   ```

3. Build swmm using cmake.

   ```
   $ cd buildprod
   $ cmake -DCMAKE_BUILD_TYPE=Release ..
   $ cmake --build . --config Release
   ```

4. Install the required python packages.

   ```
   $ pip install -r tools/requirements.txt
   ```

5. Configure and run the regression tests: where <build id>=""> - is the build identifier (i.e. swmm version number).

   ```
   $ cd ..
   $ tools/before-test.sh nrtestsuite ‘pwd‘/buildprod/bin <build id>
   $ tools/run-nrtest.sh nrtestsuite <build id>
   ```

Step by Step Guide for Windows

Coming soon ...

Working with nrtest

Using `nrtest` it is possible to compare any two versions of SWMM as long as they share the same binary file format. `nrtest` comes with a python scripts for running its `execute` and `compare` commands.

```
$ python nrtest execute apps/<app.json> tests/<test.json> -o benchmark/
$ python nrtest compare test_benchmark/ ref_benchmark/ --rtol --atol
```

To what values should rtol and atol be set?

What appears to be a simple question on the surface turns out to be more difficult upon deeper examination. The numpy testing assert allclose comparison criteria leaves a lot to be desired from a theoretical perspective, however, I have found it useful when evaluating differences between versions of SWMM.

For those wishing to dig into the details I found the linked `blog post` to be a useful starting point.

For both nrtest execute and nrtest compare non-zero exit codes are returned on failure.

More information on nrtest is available in the nrtest `docs`.

Extending the Testsuite

Where did the test and benchmark files go?

The test and benchmark files are kept in a seperate `repo`. The script before-test.sh has been provided to retreive and stage the needed files for testing.

How to add a new or different version of SWMM?

nrtest runs the command line execuatable version of SWMM for testing purposes. To add a new or different version of SWMM for benchmarking a json file that contains the absolute path to the executable location and some meta-data describing it needs to be created. See the nrtest documentation for details. The executable and the json file need to be copied to their designated locations.

How to add a test?

To add a new test the [REPORTS] section of the input file should be configured to write all results out to the binary file. A json file also needs to be created that describes how to execute the test, what files are needed, what files get generated, what comparison routines to use, and meta-data describing the test. The format and data found in the json files is resonably well described in the nrtest documentation. The input file, any auxiliary file need to run the test, and the json file describing the test should be added to the swmm-tests folder found in the swmm-example-networks repo. Finally, the test needs to be added to run-nrtest.sh so it gets called and run.

How to add a new comparison routine?

New comparison criteria can easily be implemented in the nrtest_swmm package. The package nrtest uses setup entrypoints as a simple plugin mechanism for finding comparison routines at runtime. New comparison routines can be added as a function with the swmm_xxxxxxx_compare() name pattern. The function also needs to be declared as an entry point in the nrtest_swmm setup.py file.

A basic comparison function is also provided for checking the contents of report files. It can easily be adapted for testing other types of text based files.

Common Problems

When adding new apps and tests the json format can be a bit finicky and json parsing errors aren't being reported in a user friendly manner.

The absolute path to the executable must be specified in the json file describing the application for testing.

SWMM requires the absolute path of data files referenced in a SWMM input file (e.g. rain gauge data).

The packages swmm_output and nrtest_swmm were developed and run on Windows with 64 bit version of Python 2.7. They have not been tested on Mac OS or Linux systems and have not been run using Python 3.

Unfortunately, nrtest_swmm is slow. For example comparing two 500 MB files takes on the order of 20 minutes. Therefore, tests when added should generate small output files. If your system is running a test and appears to be doing nothing it might be bogged down comparing large binary files.

Chapter 6

Stormwater-Management-Model

ORD Stormwater Management Model (aka "SWMM")

Project information

Build status

Introduction

This is the open source SWMM source code repository maintained by the Open Water Analytics group.

SWMM is a dynamic hydrology-hydraulic water quality simulation model. It is used for single event or long-term (continuous) simulation of runoff quantity and quality from primarily urban areas. SWMM source code is written in the C Programming Language and released in the Public Domain.

Contributing

Everyone is welcome to contribute to this project.

See https://github.com/OpenWaterAnalytics/Stormwater-Management-Model/blob/develop/.github/CONTRIBUTING.md "CONTRIBUTING.md" for insructions on setting your development environment.

Regression Testing

To run the test suite for please refer to the https://github.com/OpenWaterAnalytics/Stormwater-Management-Model/blob/develop/.github/REGRESSION_TESTING.md "REGRESSION_TESTING.md"

Code of conduct

The SWMM Project follows the https://github.com/OpenWaterAnalytics/Stormwater-Management-Model/blob/develop/.github/CODE_OF_CONDUCT.md "Contributor Covenant Code of Conduct"

Chapter 7

Module Index

7.1 Modules

Here is a list of all modules:

Chapter 8

Data Structure Index

8.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 9

File Index

9.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 10

Module Documentation

10.1 Authors at Open Water Analytics

The OWA-SWMM project builds on top of the contributions put forward by Lew Rossman with his original work on delivering EPA-SWMM5. We are grateful for all the work that has been put into SWMM5 and we are committed to delivering advances to the engine in a community driven approach.

Authors with contributions under the MIT-License. Authors ordered by first contribution. The authors listed here are contributors to this open source initiative and hold copyright on their contributions.

- Bryant E. McDonnell `bemcdonnell@gmail.com`
- Adam Erispaha `aerispaha@gmail.com`
- Sam Hatchett `samhatchett@gmail.com`
- Gonzalo Peña-Castellanos `goanpeca@gmail.com`
- Katherine Ratliff `ratliff.katherine@epa.gov`
- Abhiram Mullapudi `abhiramm@umich.edu`

Authors with Contributions in the Public Domain:

- Michael Tryby `tryby.michael@epa.gov`

Stand on the shoulders of Giants! For a list of all the contributors over time (SWMM5 and before), please see `Acknowledgements`

10.2 MIT-License

Contains public domain works. Other works copyright the Authors at Open Water Analytics

10.3 Network Info

Functions

- int DLLEXPORT swmm_countObjects (int type, int ∗count)

  Gets Object Count.
- int DLLEXPORT swmm_getObjectId (int type, int index, char ∗id)

  Gets Object ID.
- int DLLEXPORT swmm_getNodeType (int index, int ∗Ntype)

  Get the type of node with specified index.
- int DLLEXPORT swmm_getLinkType (int index, int ∗Ltype)

  Get the type of link with specified index.
- int DLLEXPORT swmm_getLinkConnections (int index, int ∗Node1, int ∗Node2)

  Get the link Connection Node Indeces. If the conduit has a negative slope, the dynamic wave solver will automatically reverse the nodes. To check the direction, call swmm_getLinkDirection().
- int DLLEXPORT swmm_getLinkDirection (int index, signed char ∗value)

  Get the link flow direction (see swmm_getLinkType() for notes.
- int DLLEXPORT swmm_getSubcatchOutConnection (int index, int ∗type, int ∗Index)

  Get the Subcatchment connection. Subcatchments can load to a node, another subcatchment, or itself.
- int DLLEXPORT swmm_getNodeParam (int index, int Param, double ∗value)

  Get a property value for specified node.
- int DLLEXPORT swmm_getLinkParam (int index, int Param, double ∗value)

  Get a property value for specified link.
- int DLLEXPORT swmm_getSubcatchParam (int index, int Param, double ∗value)

  Get a property value for specified subcatchment.
- int DLLEXPORT swmm_setNodeParam (int index, int Param, double value)

  Set a property value for specified node.
- int DLLEXPORT swmm_setLinkParam (int index, int Param, double value)

  Set a property value for specified link.
- int DLLEXPORT swmm_setSubcatchParam (int index, int Param, double value)

  Set a property value for specified subcatchment.

10.3.1 Detailed Description

10.3.2 Function Documentation

10.3.2.1 swmm_countObjects()

```
int swmm_countObjects (
            int type,
            int * count )
```

Gets Object Count.

Parameters

	type	Option code (see SM_ObjectType)
out	*count*	Option value

Returns

Error code

Input: type = object type (Based on SM_ObjectType enum) Output: count = pointer to integer Returns: API Error Purpose: uses Object Count table to find number of elements of an object

Definition at line 279 of file toolkitAPI.c.

10.3.2.2 swmm_getLinkConnections()

```
int swmm_getLinkConnections (
             int index,
             int * Node1,
             int * Node2 )
```

Get the link Connection Node Indeces. If the conduit has a negative slope, the dynamic wave solver will automatically reverse the nodes. To check the direction, call swmm_getLinkDirection().

Parameters

	index	The index of a link
out	*Node1*	The upstream node index.
out	*Node2*	The downstream node index.

Returns

Error code

Input: index = Index of desired ID Output: Node1 and Node2 indeces Return: API Error Purpose: Gets link Connection ID Indeces

Definition at line 427 of file toolkitAPI.c.

10.3.2.3 swmm_getLinkDirection()

```
int swmm_getLinkDirection (
             int index,
             signed char * value )
```

Get the link flow direction (see swmm_getLinkType() for notes.

Parameters

	index	The index of a link
out	*value*	The link flow direction.

Returns

Error code

Input: index = Index of desired ID Output: Link Direction (Only changes is slope $<$ 0) Return: API Error Purpose: Gets Link Direction

Definition at line 455 of file toolkitAPI.c.

10.3.2.4 swmm_getLinkParam()

```
int swmm_getLinkParam (
            int index,
            int Param,
            double * value )
```

Get a property value for specified link.

Parameters

	index	The index of a link
	Param	The property type code (See SM_LinkProperty)
out	*value*	The value of the link's property

Returns

Error code

Input: index = Index of desired ID param = Parameter desired (Based on enum SM_LinkProperty) Output: value = value to be output Return: API Error Purpose: Gets Link Parameter

Definition at line 567 of file toolkitAPI.c.

10.3.2.5 swmm_getLinkType()

```
int swmm_getLinkType (
            int index,
            int * Ltype )
```

Get the type of link with specified index.

Parameters

	index	The index of a link
out	*Ltype*	The type code for the link (SM_LinkType).

Returns

Error code

Input: index = Index of desired ID Ltype = Link type (Based on enum SM_LinkType) Return: API Error Purpose: Gets Link Type

Definition at line 403 of file toolkitAPI.c.

10.3.2.6 swmm_getNodeParam()

```
int swmm_getNodeParam (
            int index,
            int Param,
            double * value )
```

Get a property value for specified node.

Parameters

	index	The index of a node
	Param	The property type code (See SM_NodeProperty)
out	*value*	The value of the node's property

Returns

Error code

Input: index = Index of desired ID param = Parameter desired (Based on enum SM_NodeProperty) Output: value = value to be output Return: API Error Purpose: Gets Node Parameter

Definition at line 481 of file toolkitAPI.c.

10.3.2.7 swmm_getNodeType()

```
int swmm_getNodeType (
            int index,
            int * Ntype )
```

Get the type of node with specified index.

Parameters

	index	The index of a node
out	*Ntype*	The type code for the node (SM_NodeType). id must be pre-allocated by the caller.

Returns

Error code

Input: index = Index of desired ID Ntype = Node type (Based on enum SM_NodeType) Return: API Error Purpose: Gets Node Type

Definition at line 379 of file toolkitAPI.c.

10.3.2.8 swmm_getObjectId()

```
int swmm_getObjectId (
             int type,
             int index,
             char * id )
```

Gets Object ID.

Parameters

	type	Option code (see SM_ObjectType)
	index	of the Object
out	*id*	The string ID of object.

Returns

Error code

Input: type = object type (Based on SM_ObjectType enum) index = Index of desired ID Output: id = pointer to id pass by reference Return: API Error Purpose: Gets ID for any object

Definition at line 315 of file toolkitAPI.c.

10.3.2.9 swmm_getSubcatchOutConnection()

```
int swmm_getSubcatchOutConnection (
             int index,
             int * type,
             int * Index )
```

Get the Subcatchment connection. Subcatchments can load to a node, another subcatchment, or itself.

Parameters

	index	The index of a Subcatchment
out	*type*	The type of object loading (See SM_ObjectType)
out	*Index*	The object index

Returns

Error code

Input: index = Index of desired ID (Subcatchments can load to Node or another Subcatchment) Output: Type of Object Index of Object Return: API Error Purpose: Gets Subcatchment Connection ID Indeces for either Node or Subcatchment

Definition at line 760 of file toolkitAPI.c.

10.3.2.10 swmm_getSubcatchParam()

```
int swmm_getSubcatchParam (
            int index,
            int Param,
            double * value )
```

Get a property value for specified subcatchment.

Parameters

	index	The index of a subcatchment
	Param	The property type code (See SM_SubcProperty)
out	*value*	The value of the subcatchment's property

Returns

Error code

Input: index = Index of desired ID param = Parameter desired (Based on enum SM_SubcProperty) Output: value = value to be output Return: API Error Purpose: Gets Subcatchment Parameter

Definition at line 669 of file toolkitAPI.c.

10.3.2.11 swmm_setLinkParam()

```
int swmm_setLinkParam (
            int index,
            int Param,
            double value )
```

Set a property value for specified link.

Parameters

index	The index of a link
Param	The property type code (See SM_LinkProperty)
value	The new value of the link's property

Returns

Error code

Input: index = Index of desired ID param = Parameter desired (Based on enum SM_LinkProperty) value = value to be output Return: API Error Purpose: Sets Link Parameter

Definition at line 611 of file toolkitAPI.c.

10.3.2.12 swmm_setNodeParam()

```
int swmm_setNodeParam (
            int index,
            int Param,
            double value )
```

Set a property value for specified node.

Parameters

index	The index of a node
Param	The property type code (See SM_NodeProperty)
value	The new value of the node's property

Returns

Error code

Input: index = Index of desired ID param = Parameter desired (Based on enum SM_NodeProperty) value = value to be input Return: API Error Purpose: Sets Node Parameter

Definition at line 521 of file toolkitAPI.c.

10.3.2.13 swmm_setSubcatchParam()

```
int swmm_setSubcatchParam (
            int index,
            int Param,
            double value )
```

Set a property value for specified subcatchment.

Parameters

index	The index of a subcatchment
Param	The property type code (See SM_SubcProperty)
value	The new value of the subcatchment's property

Returns

Error code

Input: index = Index of desired ID param = Parameter desired (Based on enum SM_SubcProperty) value = value to be output Return: API Error Purpose: Sets Subcatchment Parameter

Definition at line 709 of file toolkitAPI.c.

10.4 Simulation Options

Functions

- void DLLEXPORT swmm_getAPIError (int errcode, char ∗s)

  Get the text of an error code.
- int DLLEXPORT swmm_getSimulationUnit (int type, int ∗value)

  Gets Simulation Unit.
- int DLLEXPORT swmm_getSimulationAnalysisSetting (int type, int ∗value)

  Gets Simulation Analysis Setting.
- int DLLEXPORT swmm_getSimulationParam (int type, double ∗value)

  Gets Simulation Analysis Setting.
- int DLLEXPORT swmm_getSimulationDateTime (int timetype, int ∗year, int ∗month, int ∗day, int ∗hour, int ∗minute, int ∗second)

  Get the current simulation datetime information.
- int DLLEXPORT swmm_setSimulationDateTime (int timetype, char ∗dtimestr)

  Set simulation datetime information.

10.4.1 Detailed Description

10.4.2 Function Documentation

10.4.2.1 swmm_getAPIError()

```
void swmm_getAPIError (
            int errcode,
            char * s )
```

Get the text of an error code.

Parameters

	errcode	The error code
out	*s*	The error string represented by the code

Input: errcode = error code Output: errmessage String Return: API Error Purpose: Get an error message

Definition at line 40 of file toolkitAPI.c.

10.4.2.2 swmm_getSimulationAnalysisSetting()

```
int swmm_getSimulationAnalysisSetting (
            int type,
            int * value )
```

Gets Simulation Analysis Setting.

Parameters

	type	Option code (see SM_SimOption)
out	*value*	Option value

Returns

Error code

Input: type = analysis type Output: setting True or False Returns: API Error Purpose: get simulation analysis setting

Definition at line 184 of file toolkitAPI.c.

10.4.2.3 swmm_getSimulationDateTime()

```
int swmm_getSimulationDateTime (
            int timetype,
            int * year,
            int * month,
            int * day,
            int * hour,
            int * minute,
            int * second )
```

Get the current simulation datetime information.

Parameters

	timetype	The property type code (See SM_TimePropety)
out	*year*	The year
out	*month*	The month
out	*day*	The day
out	*hour*	The hour
out	*minute*	The minute
out	*second*	The seconds

Returns

Error code

Input: timetype = time type to return Output: year, month, day, hours, minutes, seconds = int Return: API Error Purpose: Get the simulation start, end and report date times

Definition at line 51 of file toolkitAPI.c.

10.4.2.4 swmm_getSimulationParam()

```
int swmm_getSimulationParam (
             int type,
             double * value )
```

Gets Simulation Analysis Setting.

Parameters

	type	Option code (see SM_SimSetting)
out	*value*	Option value

Returns

Error code

Input: type = analysis type Output: Simulation Parameter Returns: error code Purpose: Get simulation analysis parameter

Definition at line 225 of file toolkitAPI.c.

10.4.2.5 swmm_getSimulationUnit()

```
int swmm_getSimulationUnit (
             int type,
             int * value )
```

Gets Simulation Unit.

Parameters

	type	Option code (see SM_Units)
out	*value*	Option value

Returns

Error code

Input: type = simulation unit type Output: enum representation of units Returns: API Error Purpose: get simulation unit types

Definition at line 152 of file toolkitAPI.c.

10.4.2.6 swmm_setSimulationDateTime()

```
swmm_setSimulationDateTime (
            int timetype,
            char * dtimestr )
```

Set simulation datetime information.

Parameters

	timetype	The property type code (See SM_TimePropety)
`out`	*dtimestr*	The current datetime. dtimestr must be pre-allocated by the caller. This will copy 19 characters.

Returns

Error code

Input: timetype = time type to return DateTime String Return: API Error Purpose: Get the simulation start, end and report date times

Definition at line 92 of file toolkitAPI.c.

10.5 Toolkit Functions

Functions

- int DLLEXPORT swmm_getCurrentDateTimeStr (char ∗dtimestr)

  Get the simulation current datetime as a string.
- int DLLEXPORT swmm_getNodeResult (int index, int type, double ∗result)

  Get a result value for specified node.
- int DLLEXPORT swmm_getLinkResult (int index, int type, double ∗result)

  Get a result value for specified link.
- int DLLEXPORT swmm_getSubcatchResult (int index, int type, double ∗result)

  Get a result value for specified subcatchment.
- int DLLEXPORT swmm_getNodeStats (int index, SM_NodeStats ∗nodeStats)

  Get a node statistics.
- int DLLEXPORT swmm_getNodeTotalInflow (int index, double ∗value)

  Get the cumulative inflow for a node.
- int DLLEXPORT swmm_getStorageStats (int index, SM_StorageStats ∗storageStats)

  Get a storage statistics.
- int DLLEXPORT swmm_getOutfallStats (int index, SM_OutfallStats ∗outfallStats)

  Get outfall statistics.
- void DLLEXPORT swmm_freeOutfallStats (SM_OutfallStats ∗outfallStats)

  Free outfall statistics structure.
- int DLLEXPORT swmm_getLinkStats (int index, SM_LinkStats ∗linkStats)

  Get link statistics.
- int DLLEXPORT swmm_getPumpStats (int index, SM_PumpStats ∗pumpStats)

  Get pump statistics.
- int DLLEXPORT swmm_getSubcatchStats (int index, SM_SubcatchStats ∗subcatchStats)

  Get subcatchment statistics.
- int DLLEXPORT swmm_getSystemRoutingStats (SM_RoutingTotals ∗routingTot)

  Get system routing statistics.
- int DLLEXPORT swmm_getSystemRunoffStats (SM_RunoffTotals ∗runoffTot)

  Get system runoff statistics.
- int DLLEXPORT swmm_setLinkSetting (int index, double setting)

  Set a link setting (pump, orifice, or weir). Setting for an orifice and a weir should be [0, 1]. A setting for a pump can range from [0, inf). However, if a pump is set to 1, it will pump at its maximum curve setting.
- int DLLEXPORT swmm_setNodeInflow (int index, double flowrate)

  Set an inflow rate to a node. The inflow rate is held constant until the caller changes it.
- int DLLEXPORT swmm_setOutfallStage (int index, double stage)

  Set outfall stage.

10.5.1 Detailed Description

10.5.2 Function Documentation

10.5.2.1 swmm_freeOutfallStats()

```
int swmm_freeOutfallStats (
            SM_OutfallStats ∗ outfallStats )
```

Free outfall statistics structure.

Parameters

out	*outfallStats*	The outfall Stats struct. This frees any allocated pollutants array.

Returns

Error code

Return: API Error Purpose: Frees Outfall Node Stats and Converts Units Note: API user is responsible for calling swmm_freeOutfallStats since this function performs a memory allocation.

Definition at line 1191 of file toolkitAPI.c.

10.5.2.2 swmm_getCurrentDateTimeStr()

```
int swmm_getCurrentDateTimeStr (
            char * dtimestr )
```

Get the simulation current datetime as a string.

Parameters

out	*dtimestr*	The current datetime. dtimestr must be pre-allocated by the caller. This will copy 19 characters.

Returns

Error code

Output: DateTime String Return: API Error Purpose: Get the current simulation time

Definition at line 808 of file toolkitAPI.c.

10.5.2.3 swmm_getLinkResult()

```
int swmm_getLinkResult (
            int index,
            int type,
            double * result )
```

Get a result value for specified link.

Parameters

	index	The index of a link
	type	The property type code (See SM_LinkResult)
out	*result*	The result of the link's property

Returns

Error code

Input: index = Index of desired ID type = Result Type (SM_LinkResult) Output: result = result data desired (byref) Return: API Error Purpose: Gets Link Simulated Value at Current Time

Definition at line 888 of file toolkitAPI.c.

10.5.2.4 swmm_getLinkStats()

```
int swmm_getLinkStats (
            int index,
            SM_LinkStats * linkStats )
```

Get link statistics.

Parameters

	index	The index of a link
out	*linkStats*	The link Stats struct (see SM_LinkStats). pre-allocated by the caller.

Returns

Error code

Output: Link Stats Structure (SM_LinkStats) Return: API Error Purpose: Gets Link Stats and Converts Units

Definition at line 1203 of file toolkitAPI.c.

10.5.2.5 swmm_getNodeResult()

```
int swmm_getNodeResult (
            int index,
            int type,
            double * result )
```

Get a result value for specified node.

Parameters

	index	The index of a node
	type	The property type code (See SM_NodeResult)
out	*result*	The result of the node's property

Returns

Error code

Input: index = Index of desired ID type = Result Type (SM_NodeResult) Output: result = result data desired (byref) Return: API Error Purpose: Gets Node Simulated Value at Current Time

Definition at line 840 of file toolkitAPI.c.

10.5.2.6 swmm_getNodeStats()

```
int swmm_getNodeStats (
            int index,
            SM_NodeStats * nodeStats )
```

Get a node statistics.

Parameters

	index	The index of a node
out	*nodeStats*	The Node Stats struct (see SM_NodeStats). pre-allocated by the caller.

Returns

Error code

Output: Node Stats Structure (SM_NodeStats) Return: API Error Purpose: Gets Node Stats and Converts Units

Definition at line 1071 of file toolkitAPI.c.

10.5.2.7 swmm_getNodeTotalInflow()

```
int swmm_getNodeTotalInflow (
            int index,
            double * value )
```

Get the cumulative inflow for a node.

Parameters

	index	The index of a node
out	*value*	The total inflow.

Returns

Error code

Input: Node Index Output: Node Total inflow Volume. Return: API Error Purpose: Get Node Total Inflow Volume.

Definition at line 1107 of file toolkitAPI.c.

10.5.2.8 swmm_getOutfallStats()

```
int swmm_getOutfallStats (
            int index,
            SM_OutfallStats * outfallStats )
```

Get outfall statistics.

Parameters

	index	The index of a outfall node
out	*outfallStats*	The outfall Stats struct (see SM_OutfallStats). pre-allocated by the caller. Caller is also responsible for freeing the SM_OutfallStats structure using swmm_freeOutfallStats(). This frees any pollutants array.

Returns

Error code

Output: Outfall Stats Structure (SM_OutfallStats) Return: API Error Purpose: Gets Outfall Node Stats and Converts Units Note: Caller is responsible for calling swmm_freeOutfallStats to free the pollutants array.

Definition at line 1152 of file toolkitAPI.c.

10.5.2.9 swmm_getPumpStats()

```
int swmm_getPumpStats (
            int index,
            SM_PumpStats * pumpStats )
```

Get pump statistics.

Parameters

	index	The index of a pump
out	*pumpStats*	The link Stats struct (see SM_PumpStats). pre-allocated by the caller.

Returns

Error code

Output: Pump Link Stats Structure (SM_PumpStats) Return: API Error Purpose: Gets Pump Link Stats and Converts Units

Definition at line 1241 of file toolkitAPI.c.

10.5.2.10 swmm_getStorageStats()

```
int swmm_getStorageStats (
            int index,
            SM_StorageStats * storageStats )
```

Get a storage statistics.

Parameters

	index	The index of a storage node
out	*storageStats*	The storage Stats struct (see SM_StorageStats). pre-allocated by the caller.

Returns

Error code

Output: Storage Node Stats Structure (SM_StorageStats) Return: API Error Purpose: Gets Storage Node Stats and Converts Units

Definition at line 1125 of file toolkitAPI.c.

10.5.2.11 swmm_getSubcatchResult()

```
int swmm_getSubcatchResult (
            int index,
            int type,
            double * result )
```

Get a result value for specified subcatchment.

Parameters

	index	The index of a subcatchment
	type	The property type code (See SM_SubcResult)
out	*result*	The result of the subcatchment's property

Returns

Error code

Input: index = Index of desired ID type = Result Type (SM_SubcResult) Output: result = result data desired (byref) Return: API Error Purpose: Gets Subcatchment Simulated Value at Current Time

Definition at line 935 of file toolkitAPI.c.

10.5.2.12 swmm_getSubcatchStats()

```
int swmm_getSubcatchStats (
            int index,
            SM_SubcatchStats * subcatchStats )
```

Get subcatchment statistics.

Parameters

	index	The index of a subcatchment
out	*subcatchStats*	The link Stats struct (see SM_SubcatchStats). pre-allocated by the caller. Caller is also responsible for freeing the SM_SubcatchStats structure using swmm_freeSubcatchStats(). This frees any pollutants array.

Returns

Error code

Output: Subcatchment Stats Structure (SM_SubcatchStats) Return: API Error Purpose: Gets Subcatchment Stats and Converts Units

Definition at line 1272 of file toolkitAPI.c.

10.5.2.13 swmm_getSystemRoutingStats()

```
int swmm_getSystemRoutingStats (
            SM_RoutingTotals * routingTot )
```

Get system routing statistics.

Parameters

out	*routingTot*	The system Routing Stats struct (see SM_RoutingTotals). pre-allocated by the caller.

Returns

Error code

Output: System Routing Totals Structure (SM_RoutingTotals) Return: API Error Purpose: Gets System Flow Routing Totals and Converts Units

Definition at line 1302 of file toolkitAPI.c.

10.5.2.14 swmm_getSystemRunoffStats()

```
int swmm_getSystemRunoffStats (
            SM_RunoffTotals * runoffTot )
```

Get system runoff statistics.

Parameters

out	*runoffTot*	The system Runoff Stats struct (see SM_RunoffTotals). pre-allocated by the caller.

Returns

Error code

Output: System Runoff Totals Structure (SM_RunoffTotals) Return: API Error Purpose: Gets System Runoff Totals and Converts Units

Definition at line 1337 of file toolkitAPI.c.

10.5.2.15 swmm_setLinkSetting()

```
int swmm_setLinkSetting (
            int index,
            double setting )
```

Set a link setting (pump, orifice, or weir). Setting for an orifice and a weir should be [0, 1]. A setting for a pump can range from [0, inf). However, if a pump is set to 1, it will pump at its maximum curve setting.

Parameters

index	The link index.
setting	The new setting for the link.

Returns

Error code

Input: index = Index of desired ID value = New Target Setting Output: returns API Error Purpose: Sets Link open fraction (Weir, Orifice, Pump, and Outlet)

Definition at line 1377 of file toolkitAPI.c.

10.5.2.16 swmm_setNodeInflow()

```
int swmm_setNodeInflow (
            int index,
            double flowrate )
```

Set an inflow rate to a node. The inflow rate is held constant until the caller changes it.

Parameters

index	The node index.
flowrate	The new node inflow rate.

Returns

Error code

Input: index = Index of desired ID value = New Inflow Rate Output: returns API Error Purpose: Sets new node inflow rate and holds until set again

Definition at line 1420 of file toolkitAPI.c.

10.5.2.17 swmm_setOutfallStage()

```
int swmm_setOutfallStage (
            int index,
            double stage )
```

Set outfall stage.

Parameters

index	The outfall node index.
stage	The outfall node stage (head).

Returns

Error code

Input: index = Index of desired outfall stage = New outfall stage (head) Output: returns API Error Purpose: Sets new outfall stage and holds until set again.

Definition at line 1481 of file toolkitAPI.c.

Chapter 11

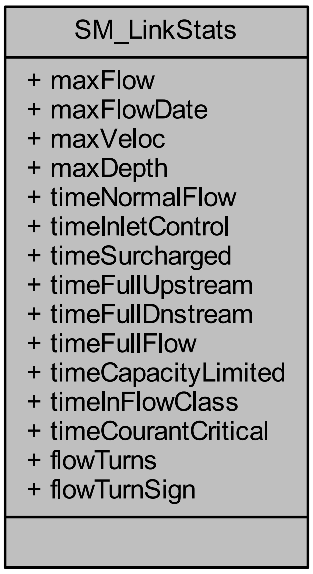
Data Structure Documentation

11.1 SM_LinkStats Struct Reference

Link stats structure.

`#include <toolkitAPI.h>`

Collaboration diagram for SM_LinkStats:



11.1.1 Detailed Description

Link stats structure.

Definition at line 235 of file toolkitAPI.h.

The documentation for this struct was generated from the following file:

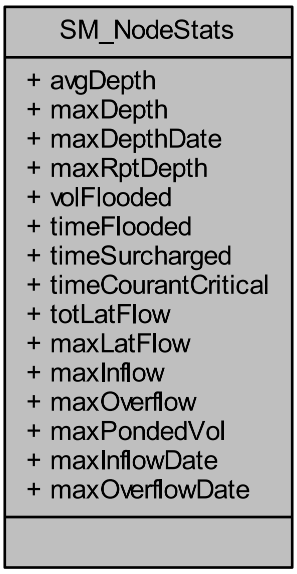
- C:/PROJECTCODE/Stormwater-Management-Model/include/toolkitAPI.h

11.2 SM_NodeStats Struct Reference

Node stats structure.

```
#include <toolkitAPI.h>
```

Collaboration diagram for SM_NodeStats:



11.2.1 Detailed Description

Node stats structure.

Definition at line 194 of file toolkitAPI.h.

The documentation for this struct was generated from the following file:

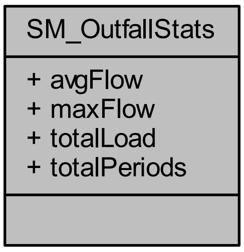
- C:/PROJECTCODE/Stormwater-Management-Model/include/toolkitAPI.h

11.3 SM_OutfallStats Struct Reference

Outfall stats structure.

```
#include <toolkitAPI.h>
```

Collaboration diagram for SM_OutfallStats:



11.3.1 Detailed Description

Outfall stats structure.

Definition at line 226 of file toolkitAPI.h.

The documentation for this struct was generated from the following file:

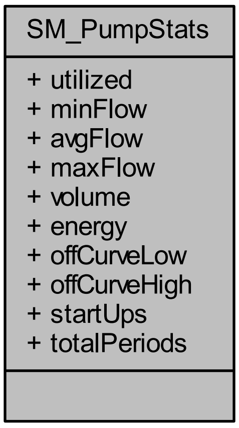
- C:/PROJECTCODE/Stormwater-Management-Model/include/toolkitAPI.h

11.4 SM_PumpStats Struct Reference

Pump stats structure.

```
#include <toolkitAPI.h>
```

Collaboration diagram for SM_PumpStats:



11.4.1 Detailed Description

Pump stats structure.

Definition at line 255 of file toolkitAPI.h.

The documentation for this struct was generated from the following file:

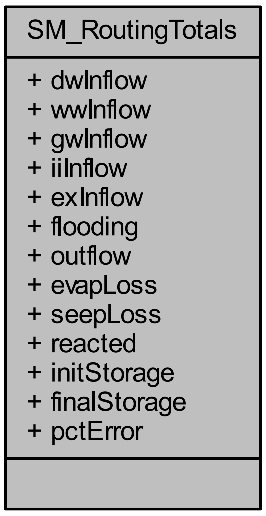
- C:/PROJECTCODE/Stormwater-Management-Model/include/toolkitAPI.h

11.5 SM_RoutingTotals Struct Reference

System routing stats structure.

```
#include <toolkitAPI.h>
```

Collaboration diagram for SM_RoutingTotals:



11.5.1 Detailed Description

System routing stats structure.

Definition at line 281 of file toolkitAPI.h.

The documentation for this struct was generated from the following file:

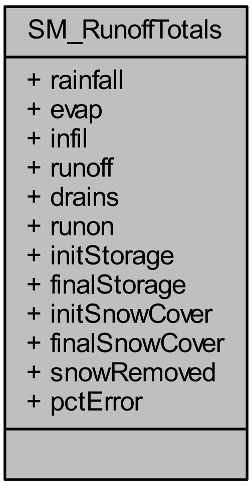
- C:/PROJECTCODE/Stormwater-Management-Model/include/toolkitAPI.h

11.6 SM_RunoffTotals Struct Reference

System runoff stats structure.

```
#include <toolkitAPI.h>
```

Collaboration diagram for SM_RunoffTotals:



11.6.1 Detailed Description

System runoff stats structure.

Definition at line 299 of file toolkitAPI.h.

The documentation for this struct was generated from the following file:

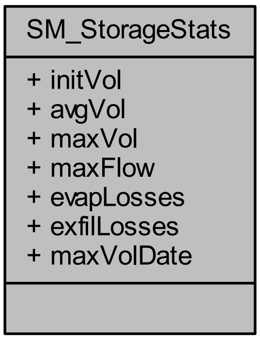
- C:/PROJECTCODE/Stormwater-Management-Model/include/toolkitAPI.h

11.7 SM_StorageStats Struct Reference

Storage stats structure.

```
#include <toolkitAPI.h>
```

Collaboration diagram for SM_StorageStats:



11.7.1 Detailed Description

Storage stats structure.

Definition at line 214 of file toolkitAPI.h.

The documentation for this struct was generated from the following file:

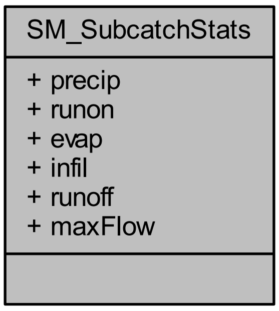
- C:/PROJECTCODE/Stormwater-Management-Model/include/toolkitAPI.h

11.8 SM_SubcatchStats Struct Reference

Subcatchment stats structure.

```
#include <toolkitAPI.h>
```

Collaboration diagram for SM_SubcatchStats:



11.8.1 Detailed Description

Subcatchment stats structure.

Definition at line 270 of file toolkitAPI.h.

The documentation for this struct was generated from the following file:

- C:/PROJECTCODE/Stormwater-Management-Model/include/toolkitAPI.h

Chapter 12

File Documentation

12.1 C:/PROJECTCODE/Stormwater-Management-Model/include/swmm5.h File Reference

Prototypes for SWMM5 functions exported to swmm5.dll.

Functions

- int DLLEXPORT swmm_run (char ∗f1, char ∗f2, char ∗f3)

    Opens SWMM input file, reads in network data, runs, and closes.
- int DLLEXPORT swmm_open (char ∗f1, char ∗f2, char ∗f3)

    Opens SWMM input file & reads in network data.
- int DLLEXPORT swmm_start (int saveFlag)

    Start SWMM simulation.
- int DLLEXPORT swmm_step (double ∗elapsedTime)

    Step SWMM simulation forward.
- int DLLEXPORT swmm_end (void)

    End SWMM simulation.
- int DLLEXPORT swmm_report (void)

    Write text report file.
- int DLLEXPORT swmm_getMassBalErr (float ∗runoffErr, float ∗flowErr, float ∗qualErr)

    Get routing errors.
- int DLLEXPORT swmm_close (void)

    Frees all memory and files used by SWMM.
- int DLLEXPORT swmm_getVersion (void)

    Get Legacy SWMM version number.
- void DLLEXPORT swmm_getSemVersion (char ∗semver)

    Get full semantic version number.
- void DLLEXPORT swmm_getVersionInfo (char ∗major, char ∗minor, char ∗patch)

    Get full semantic version number info.

12.1.1 Detailed Description

Prototypes for SWMM5 functions exported to swmm5.dll.

See also

`http://github.com/openwateranalytics/stormwater-management-model`

swmm5.h

Date

03/24/14 (Build 5.1.001)
08/01/16 (Build 5.1.011)

Version

5.1

Authors

L. Rossman, OpenWaterAnalytics members: see `AUTHORS`.

12.1.2 Function Documentation

12.1.2.1 swmm_close()

```
int DLLEXPORT swmm_close (
             void   )
```

Frees all memory and files used by SWMM.

Returns

Error code

Definition at line 685 of file swmm5.c.

12.1.2.2 swmm_end()

```
int DLLEXPORT swmm_end (
             void   )
```

End SWMM simulation.

Returns

error code

Definition at line 626 of file swmm5.c.

12.1.2.3 swmm_getMassBalErr()

```
int DLLEXPORT swmm_getMassBalErr (
            float * runoffErr,
            float * flowErr,
            float * qualErr )
```

Get routing errors.

Parameters

out	*runoffErr*	Runoff routing error
out	*flowErr*	Flow routing error
out	*qualErr*	Quality routing error

Returns

error code

Definition at line 709 of file swmm5.c.

12.1.2.4 swmm_getSemVersion()

```
void DLLEXPORT swmm_getSemVersion (
            char * semver )
```

Get full semantic version number.

Parameters

out	*semver*	sematic version (char array)

Definition at line 749 of file swmm5.c.

12.1.2.5 swmm_getVersion()

```
int DLLEXPORT swmm_getVersion (
            void )
```

Get Legacy SWMM version number.

Returns

Version

Definition at line 735 of file swmm5.c.

12.1.2.6 swmm_getVersionInfo()

```
void DLLEXPORT swmm_getVersionInfo (
            char * major,
            char * minor,
            char * patch )
```

Get full semantic version number info.

Parameters

out	*major*	sematic version major number
out	*minor*	sematic version minor number
out	*patch*	sematic version patch number

Definition at line 759 of file swmm5.c.

12.1.2.7 swmm_open()

```
int DLLEXPORT swmm_open (
            char * f1,
            char * f2,
            char * f3 )
```

Opens SWMM input file & reads in network data.

Parameters

f1	pointer to name of input file (must exist)
f2	pointer to name of report file (to be created)
f3	pointer to name of binary output file (to be created)

Returns

error code

Definition at line 348 of file swmm5.c.

12.1.2.8 swmm_report()

```
int DLLEXPORT swmm_report (
            void )
```

Write text report file.

Returns

error code

Definition at line 666 of file swmm5.c.

12.1.2.9 swmm_run()

```
int DLLEXPORT swmm_run (
            char * f1,
            char * f2,
            char * f3 )
```

Opens SWMM input file, reads in network data, runs, and closes.

Parameters

f1	pointer to name of input file (must exist)
f2	pointer to name of report file (to be created)
f3	pointer to name of binary output file (to be created)

Returns

error code

Definition at line 287 of file swmm5.c.

12.1.2.10 swmm_start()

```
int DLLEXPORT swmm_start (
              int saveFlag )
```

Start SWMM simulation.

Parameters

saveFlag	TRUE or FALSE to save timeseries to report file

Returns

error code

Definition at line 410 of file swmm5.c.

12.1.2.11 swmm_step()

```
int DLLEXPORT swmm_step (
              double * elapsedTime )
```

Step SWMM simulation forward.

Parameters

out	*elapsedTime*	elapsed simulation time [milliseconds]

Returns

error code

Definition at line 500 of file swmm5.c.

12.2 C:/PROJECTCODE/Stormwater-Management-Model/include/toolkitAPI.h File Reference

Exportable Functions for Toolkit API.

```
#include "../src/datetime.h"
```

Data Structures

- struct SM_NodeStats

  Node stats structure.
- struct SM_StorageStats

  Storage stats structure.
- struct SM_OutfallStats

  Outfall stats structure.
- struct SM_LinkStats

  Link stats structure.
- struct SM_PumpStats

  Pump stats structure.
- struct SM_SubcatchStats

  Subcatchment stats structure.
- struct SM_RoutingTotals

  System routing stats structure.
- struct SM_RunoffTotals

  System runoff stats structure.

Enumerations

- enum SM_ObjectType {
  SM_GAGE = 0, SM_SUBCATCH = 1, SM_NODE = 2, SM_LINK = 3,
  SM_POLLUT = 4, SM_LANDUSE = 5, SM_TIMEPATTERN = 6, SM_CURVE = 7,
  SM_TSERIES = 8, SM_CONTROL = 9, SM_TRANSECT = 10, SM_AQUIFER = 11,
  SM_UNITHYD = 12, SM_SNOWMELT = 13, SM_SHAPE = 14, SM_LID = 15 }

  Object type codes.
- enum SM_NodeType { SM_JUNCTION = 0, SM_OUTFALL = 1, SM_STORAGE = 2, SM_DIVIDER = 3 }

  Node object type codes.
- enum SM_LinkType {
  SM_CONDUIT = 0, SM_PUMP = 1, SM_ORIFICE = 2, SM_WEIR = 3,
  SM_OUTLET = 4 }

  Link object type codes.
- enum SM_TimePropety { SM_STARTDATE = 0, SM_ENDDATE = 1, SM_REPORTDATE = 2 }

  Simulation Option codes.
- enum SM_Units { SM_SYSTEMUNIT = 0, SM_FLOWUNIT = 1 }

  Simulation Unit Codes.
- enum SM_SimOption {
  SM_ALLOWPOND = 0, SM_SKIPSTEADY = 1, SM_IGNORERAIN = 2, SM_IGNORERDII = 3,
  SM_IGNORESNOW = 4, SM_IGNOREGW = 5, SM_IGNOREROUTE = 6, SM_IGNORERQUAL = 7 }

  Simulation Options.

- enum SM_SimSetting {
SM_ROUTESTEP = 0, SM_MINROUTESTEP = 1, SM_LENGTHSTEP = 2, SM_STARTDRYDAYS = 3,
SM_COURANTFACTOR = 4, SM_MINSURFAREA = 5, SM_MINSLOPE = 6, SM_RUNOFFERROR = 7,
SM_GWERROR = 8, SM_FLOWERROR = 9, SM_QUALERROR = 10, SM_HEADTOL = 11,
SM_SYSFLOWTOL = 12, SM_LATFLOWTOL = 13 }

  Simulation Settings.
- enum SM_NodeProperty {
SM_INVERTEL = 0, SM_FULLDEPTH = 1, SM_SURCHDEPTH = 2, SM_PONDAREA = 3,
SM_INITDEPTH = 4 }

  Node property codes.
- enum SM_LinkProperty {
SM_OFFSET1 = 0, SM_OFFSET2 = 1, SM_INITFLOW = 2, SM_FLOWLIMIT = 3,
SM_INLETLOSS = 4, SM_OUTLETLOSS = 5, SM_AVELOSS = 6 }

  Link property codes.
- enum SM_SubcProperty {
SM_WIDTH = 0, SM_AREA = 1, SM_FRACIMPERV = 2, SM_SLOPE = 3,
SM_CURBLEN = 4 }

  Subcatchment property codes.
- enum SM_NodeResult {
SM_TOTALINFLOW = 0, SM_TOTALOUTFLOW = 1, SM_LOSSES = 2, SM_NODEVOL = 3,
SM_NODEFLOOD = 4, SM_NODEDEPTH = 5, SM_NODEHEAD = 6, SM_LATINFLOW = 7 }

  Node result property codes.
- enum SM_LinkResult {
SM_LINKFLOW = 0, SM_LINKDEPTH = 1, SM_LINKVOL = 2, SM_USSURFAREA = 3,
SM_DSSURFAREA = 4, SM_SETTING = 5, SM_TARGETSETTING = 6, SM_FROUDE = 7 }

  Link result property codes.
- enum SM_SubcResult {
SM_SUBCRAIN = 0, SM_SUBCEVAP = 1, SM_SUBCINFIL = 2, SM_SUBCRUNON = 3,
SM_SUBCRUNOFF = 4, SM_SUBCSNOW = 5 }

  Subcatchment result property codes.
- enum SM_SubcPollut { SM_BUILDUP = 0, SM_CPONDED = 1 }

  Subcatchment pollutant result property codes.
- enum SM_GagePrecip { SM_TOTALPRECIP = 0, SM_RAINFALL = 1, SM_SNOWFALL = 2 }

  Gage precip array property codes.

Functions

- void DLLEXPORT swmm_getAPIError (int errcode, char ∗s)

  Get the text of an error code.
- int DLLEXPORT swmm_getSimulationUnit (int type, int ∗value)

  Gets Simulation Unit.
- int DLLEXPORT swmm_getSimulationAnalysisSetting (int type, int ∗value)

  Gets Simulation Analysis Setting.
- int DLLEXPORT swmm_getSimulationParam (int type, double ∗value)

  Gets Simulation Analysis Setting.
- int DLLEXPORT swmm_countObjects (int type, int ∗count)

  Gets Object Count.
- int DLLEXPORT swmm_getObjectId (int type, int index, char ∗id)

  Gets Object ID.
- int DLLEXPORT swmm_getObjectIndex (int type, char ∗id, int ∗errcode)

  Gets Object ID Index.
- int DLLEXPORT swmm_getNodeType (int index, int ∗Ntype)

Get the type of node with specified index.

- int DLLEXPORT swmm_getLinkType (int index, int ∗Ltype)

  Get the type of link with specified index.
- int DLLEXPORT swmm_getLinkConnections (int index, int ∗Node1, int ∗Node2)

  Get the link Connection Node Indeces. If the conduit has a negative slope, the dynamic wave solver will automatically reverse the nodes. To check the direction, call swmm_getLinkDirection().
- int DLLEXPORT swmm_getLinkDirection (int index, signed char ∗value)

  Get the link flow direction (see swmm_getLinkType() for notes.
- int DLLEXPORT swmm_getSubcatchOutConnection (int index, int ∗type, int ∗Index)

  Get the Subcatchment connection. Subcatchments can load to a node, another subcatchment, or itself.
- int DLLEXPORT swmm_getNodeParam (int index, int Param, double ∗value)

  Get a property value for specified node.
- int DLLEXPORT swmm_setNodeParam (int index, int Param, double value)

  Set a property value for specified node.
- int DLLEXPORT swmm_getLinkParam (int index, int Param, double ∗value)

  Get a property value for specified link.
- int DLLEXPORT swmm_setLinkParam (int index, int Param, double value)

  Set a property value for specified link.
- int DLLEXPORT swmm_getSubcatchParam (int index, int Param, double ∗value)

  Get a property value for specified subcatchment.
- int DLLEXPORT swmm_setSubcatchParam (int index, int Param, double value)

  Set a property value for specified subcatchment.
- int DLLEXPORT swmm_getSimulationDateTime (int timetype, int ∗year, int ∗month, int ∗day, int ∗hour, int ∗minute, int ∗second)

  Get the current simulation datetime information.
- int DLLEXPORT swmm_setSimulationDateTime (int timetype, char ∗dtimestr)

  Set simulation datetime information.
- int DLLEXPORT swmm_getCurrentDateTimeStr (char ∗dtimestr)

  Get the simulation current datetime as a string.
- int DLLEXPORT swmm_getNodeResult (int index, int type, double ∗result)

  Get a result value for specified node.
- int DLLEXPORT swmm_getLinkResult (int index, int type, double ∗result)

  Get a result value for specified link.
- int DLLEXPORT swmm_getSubcatchResult (int index, int type, double ∗result)

  Get a result value for specified subcatchment.
- int DLLEXPORT swmm_getSubcatchPollut (int index, int type, double ∗∗PollutArray)

  Gets pollutant values for a specified subcatchment.
- int DLLEXPORT swmm_getGagePrecip (int index, double ∗∗GageArray)

  Get precipitation rates for a gage.
- int DLLEXPORT swmm_getNodeStats (int index, SM_NodeStats ∗nodeStats)

  Get a node statistics.
- int DLLEXPORT swmm_getNodeTotalInflow (int index, double ∗value)

  Get the cumulative inflow for a node.
- int DLLEXPORT swmm_getStorageStats (int index, SM_StorageStats ∗storageStats)

  Get a storage statistics.
- int DLLEXPORT swmm_getOutfallStats (int index, SM_OutfallStats ∗outfallStats)

  Get outfall statistics.
- void DLLEXPORT swmm_freeOutfallStats (SM_OutfallStats ∗outfallStats)

  Free outfall statistics structure.
- int DLLEXPORT swmm_getLinkStats (int index, SM_LinkStats ∗linkStats)

  Get link statistics.

- int DLLEXPORT swmm_getPumpStats (int index, SM_PumpStats ∗pumpStats)

  Get pump statistics.
- int DLLEXPORT swmm_getSubcatchStats (int index, SM_SubcatchStats ∗subcatchStats)

  Get subcatchment statistics.
- int DLLEXPORT swmm_getSystemRoutingStats (SM_RoutingTotals ∗routingTot)

  Get system routing statistics.
- int DLLEXPORT swmm_getSystemRunoffStats (SM_RunoffTotals ∗runoffTot)

  Get system runoff statistics.
- int DLLEXPORT swmm_setLinkSetting (int index, double setting)

  Set a link setting (pump, orifice, or weir). Setting for an orifice and a weir should be [0, 1]. A setting for a pump can range from [0, inf). However, if a pump is set to 1, it will pump at its maximum curve setting.
- int DLLEXPORT swmm_setNodeInflow (int index, double flowrate)

  Set an inflow rate to a node. The inflow rate is held constant until the caller changes it.
- int DLLEXPORT swmm_setOutfallStage (int index, double stage)

  Set outfall stage.
- int DLLEXPORT swmm_setGagePrecip (int index, double total_precip)

  Set a total precipitation intensity to the gage.
- void DLLEXPORT freeArray (void ∗∗array)

  Helper function to free memory array allocated in SWMM.

12.2.1 Detailed Description

Exportable Functions for Toolkit API.

See also

`http://github.com/openwateranalytics/stormwater-management-model`

toolkitAPI.h

Date

08/30/2016 (First Contribution)

Authors

B. McDonnell (EmNet LLC), OpenWaterAnalytics members: see `AUTHORS`.

12.2.2 Enumeration Type Documentation

12.2.2.1 SM_GagePrecip

`enum SM_GagePrecip`

Gage precip array property codes.

Enumerator

SM_TOTALPRECIP	Total Precipitation Rate
SM_RAINFALL	Rainfall Rate
SM_SNOWFALL	Snowfall Rate

Definition at line 185 of file toolkitAPI.h.

12.2.2.2 SM_LinkProperty

```
enum SM_LinkProperty
```

Link property codes.

Enumerator

SM_OFFSET1	Inlet Offset
SM_OFFSET2	Outlet Offset
SM_INITFLOW	Initial Flow Rate
SM_FLOWLIMIT	Flow limit
SM_INLETLOSS	Inlet Loss
SM_OUTLETLOSS	Outles Loss
SM_AVELOSS	Average Loss

Definition at line 125 of file toolkitAPI.h.

12.2.2.3 SM_LinkResult

```
enum SM_LinkResult
```

Link result property codes.

Enumerator

SM_LINKFLOW	Flowrate
SM_LINKDEPTH	Depth
SM_LINKVOL	Volume
SM_USSURFAREA	Upstream Surface Area
SM_DSSURFAREA	Downstream Surface Area
SM_SETTING	Setting
SM_TARGETSETTING	Target Setting
SM_FROUDE	Froude Number

Definition at line 157 of file toolkitAPI.h.

12.2.2.4 SM_LinkType

```
enum SM_LinkType
```

Link object type codes.

Enumerator

SM_CONDUIT	Conduit
SM_PUMP	Pump
SM_ORIFICE	Orifice
SM_WEIR	Weir
SM_OUTLET	Outlet

Definition at line 64 of file toolkitAPI.h.

12.2.2.5 SM_NodeProperty

```
enum SM_NodeProperty
```

Node property codes.

Enumerator

SM_INVERTEL	Invert Elevation
SM_FULLDEPTH	Full Depth
SM_SURCHDEPTH	Surcharge Depth
SM_PONDAREA	Ponding Area
SM_INITDEPTH	Initial Depth

Definition at line 116 of file toolkitAPI.h.

12.2.2.6 SM_NodeResult

```
enum SM_NodeResult
```

Node result property codes.

Enumerator

SM_TOTALINFLOW	Total Inflow

Enumerator

SM_TOTALOUTFLOW	Total Outflow
SM_LOSSES	Node Losses
SM_NODEVOL	Stored Volume
SM_NODEFLOOD	Flooding Rate
SM_NODEDEPTH	Node Depth
SM_NODEHEAD	Node Head
SM_LATINFLOW	Lateral Inflow Rate

Definition at line 145 of file toolkitAPI.h.

12.2.2.7 SM_NodeType

```
enum SM_NodeType
```

Node object type codes.

Enumerator

SM_JUNCTION	Manhole Junction
SM_OUTFALL	Outfall
SM_STORAGE	Storage
SM_DIVIDER	Divider

Definition at line 56 of file toolkitAPI.h.

12.2.2.8 SM_ObjectType

```
enum SM_ObjectType
```

Object type codes.

Enumerator

SM_GAGE	Rain gage
SM_SUBCATCH	Subcatchment
SM_NODE	Conveyance system node
SM_LINK	Conveyance system link
SM_POLLUT	Pollutant
SM_LANDUSE	Land use category
SM_TIMEPATTERN	Dry weather flow time pattern
SM_CURVE	Generic table of values
SM_TSERIES	Generic time series of values
SM_CONTROL	Conveyance system control rules

Enumerator

SM_TRANSECT	Irregular channel cross-section
SM_AQUIFER	Groundwater aquifer
SM_UNITHYD	RDII unit hydrograph
SM_SNOWMELT	Snowmelt parameter set
SM_SHAPE	Custom conduit shape
SM_LID	LID treatment units

Definition at line 36 of file toolkitAPI.h.

12.2.2.9 SM_SimOption

```
enum SM_SimOption
```

Simulation Options.

Enumerator

SM_ALLOWPOND	Allow Ponding
SM_SKIPSTEADY	Skip Steady State
SM_IGNORERAIN	Ignore Rainfall
SM_IGNORERDII	Ignore RDII
SM_IGNORESNOW	Ignore Snowmelt
SM_IGNOREGW	Ignore Groundwater
SM_IGNOREROUTE	Ignore Routing
SM_IGNORERQUAL	Ignore Quality

Definition at line 86 of file toolkitAPI.h.

12.2.2.10 SM_SimSetting

```
enum SM_SimSetting
```

Simulation Settings.

Enumerator

SM_ROUTESTEP	Routing Step (sec)
SM_MINROUTESTEP	Minimum Routing Step (sec)
SM_LENGTHSTEP	Lengthening Step (sec)
SM_STARTDRYDAYS	Antecedent dry days
SM_COURANTFACTOR	Courant time step factor
SM_MINSURFAREA	Minimum nodal surface area

Enumerator

SM_MINSLOPE	Minimum conduit slope
SM_RUNOFFERROR	Runoff continuity error
SM_GWERROR	Groundwater continuity error
SM_FLOWERROR	Flow routing error
SM_QUALERROR	Quality routing error
SM_HEADTOL	DW routing head tolerance (ft)
SM_SYSFLOWTOL	Tolerance for steady system flow
SM_LATFLOWTOL	Tolerance for steady nodal inflow

Definition at line 98 of file toolkitAPI.h.

12.2.2.11 SM_SubcPollut

```
enum SM_SubcPollut
```

Subcatchment pollutant result property codes.

Enumerator

SM_BUILDUP	Pollutant Buildup Load
SM_CPONDED	Ponded Pollutant Concentration

Definition at line 179 of file toolkitAPI.h.

12.2.2.12 SM_SubcProperty

```
enum SM_SubcProperty
```

Subcatchment property codes.

Enumerator

SM_WIDTH	Width
SM_AREA	Area
SM_FRACIMPERV	Impervious Fraction
SM_SLOPE	Slope
SM_CURBLEN	Curb Length

Definition at line 136 of file toolkitAPI.h.

12.2.2.13 SM_SubcResult

```
enum SM_SubcResult
```

Subcatchment result property codes.

Enumerator

SM_SUBCRAIN	Rainfall Rate
SM_SUBCEVAP	Evaporation Loss
SM_SUBCINFIL	Infiltration Loss
SM_SUBCRUNON	Runon Rate
SM_SUBCRUNOFF	Runoff Rate
SM_SUBCSNOW	Snow Depth

Definition at line 169 of file toolkitAPI.h.

12.2.2.14 SM_TimePropety

```
enum SM_TimePropety
```

Simulation Option codes.

Enumerator

SM_STARTDATE	Simulation Start Date
SM_ENDDATE	Simulation End Date
SM_REPORTDATE	Simulation Report Start Date

Definition at line 73 of file toolkitAPI.h.

12.2.2.15 SM_Units

```
enum SM_Units
```

Simulation Unit Codes.

Enumerator

SM_SYSTEMUNIT	System Units
SM_FLOWUNIT	Flow Units

Definition at line 80 of file toolkitAPI.h.

12.2.3 Function Documentation

12.2.3.1 freeArray()

```
void DLLEXPORT freeArray (
            void ** array )
```

Helper function to free memory array allocated in SWMM.

Parameters

array	The pointer to the array

Helper function used to free array allocated memory by API.

Definition at line 1567 of file toolkitAPI.c.

12.2.3.2 swmm_getGagePrecip()

```
int DLLEXPORT swmm_getGagePrecip (
            int index,
            double ** GageArray )
```

Get precipitation rates for a gage.

Parameters

	index	The index of gage
out	*GageArray*	precipitation rates array [total, rainfall, snowfall]

Returns

Error code

Input: index = Index of desired ID Output: GageArray pointer (three elements) Return: API Error Purpose: Gets the precipitation value in the gage.

Definition at line 1032 of file toolkitAPI.c.

12.2.3.3 swmm_getObjectIndex()

```
int DLLEXPORT swmm_getObjectIndex (
            int type,
            char * id,
            int * errcode )
```

Gets Object ID Index.

Parameters

	type	Option code (see SM_ObjectType)
out	*id*	of the Object
out	*errcode*	Error Code

Returns

Object Injdex

Input: type = object type (Based on SM_ObjectType enum) char∗ = ID name Output: errorcode = pointer to error code Return: Object Index Purpose: Gets object id index

Definition at line 292 of file toolkitAPI.c.

12.2.3.4 swmm_getSubcatchPollut()

```
int DLLEXPORT swmm_getSubcatchPollut (
             int index,
             int type,
             double ** PollutArray )
```

Gets pollutant values for a specified subcatchment.

Parameters

	index	The index of a subcatchment
	type	The property type code (see SM_SubcPollut)
out	*PollutArray*	result array

Returns

Error code

Input: index = Index of desired ID type = Result Type (SM_SubcPollut) Output: PollutArray pointer (pollutant data desired, byref) Return: API Error Purpose: Gets Subcatchment Simulated Pollutant Value at Current Time

Definition at line 978 of file toolkitAPI.c.

12.2.3.5 swmm_setGagePrecip()

```
int DLLEXPORT swmm_setGagePrecip (
             int index,
             double total_precip )
```

Set a total precipitation intensity to the gage.

Parameters

index	The gage index.
total_precip	The new total precipitation intensity.

Returns

Error code

Input: index = Index of desired ID total_precip = rainfall intensity to be set Return: API Error Purpose: Sets the precipitation in from the external database

Definition at line 1516 of file toolkitAPI.c.

12.3 C:/PROJECTCODE/Stormwater-Management-Model/src/toolkitAPI.c File Reference

Exportable Functions for Toolkit API.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "headers.h"
#include "swmm5.h"
#include "hash.h"
```

Functions

- double ∗ newDoubleArray (int n)
- void DLLEXPORT swmm_getAPIError (int errcode, char ∗s)

  Get the text of an error code.
- int DLLEXPORT swmm_getSimulationDateTime (int timetype, int ∗year, int ∗month, int ∗day, int ∗hour, int ∗minute, int ∗second)

  Get the current simulation datetime information.
- int DLLEXPORT swmm_setSimulationDateTime (int timetype, char ∗dtimestr)

  Set simulation datetime information.
- int DLLEXPORT swmm_getSimulationUnit (int type, int ∗value)

  Gets Simulation Unit.
- int DLLEXPORT swmm_getSimulationAnalysisSetting (int type, int ∗value)

  Gets Simulation Analysis Setting.
- int DLLEXPORT swmm_getSimulationParam (int type, double ∗value)

  Gets Simulation Analysis Setting.
- int DLLEXPORT swmm_countObjects (int type, int ∗count)

  Gets Object Count.
- int DLLEXPORT swmm_getObjectIndex (int type, char ∗id, int ∗errcode)

  Gets Object ID Index.
- int DLLEXPORT swmm_getObjectId (int type, int index, char ∗id)

  Gets Object ID.
- int DLLEXPORT swmm_getNodeType (int index, int ∗Ntype)

Get the type of node with specified index.
- int DLLEXPORT swmm_getLinkType (int index, int ∗Ltype)

  Get the type of link with specified index.
- int DLLEXPORT swmm_getLinkConnections (int index, int ∗Node1, int ∗Node2)

  Get the link Connection Node Indeces. If the conduit has a negative slope, the dynamic wave solver will automatically reverse the nodes. To check the direction, call swmm_getLinkDirection().
- int DLLEXPORT swmm_getLinkDirection (int index, signed char ∗value)

  Get the link flow direction (see swmm_getLinkType() for notes.
- int DLLEXPORT swmm_getNodeParam (int index, int Param, double ∗value)

  Get a property value for specified node.
- int DLLEXPORT swmm_setNodeParam (int index, int Param, double value)

  Set a property value for specified node.
- int DLLEXPORT swmm_getLinkParam (int index, int Param, double ∗value)

  Get a property value for specified link.
- int DLLEXPORT swmm_setLinkParam (int index, int Param, double value)

  Set a property value for specified link.
- int DLLEXPORT swmm_getSubcatchParam (int index, int Param, double ∗value)

  Get a property value for specified subcatchment.
- int DLLEXPORT swmm_setSubcatchParam (int index, int Param, double value)

  Set a property value for specified subcatchment.
- int DLLEXPORT swmm_getSubcatchOutConnection (int index, int ∗type, int ∗Index)

  Get the Subcatchment connection. Subcatchments can load to a node, another subcatchment, or itself.
- int DLLEXPORT swmm_getCurrentDateTimeStr (char ∗dtimestr)

  Get the simulation current datetime as a string.
- int DLLEXPORT swmm_getNodeResult (int index, int type, double ∗result)

  Get a result value for specified node.
- int DLLEXPORT swmm_getLinkResult (int index, int type, double ∗result)

  Get a result value for specified link.
- int DLLEXPORT swmm_getSubcatchResult (int index, int type, double ∗result)

  Get a result value for specified subcatchment.
- int DLLEXPORT swmm_getSubcatchPollut (int index, int type, double ∗∗PollutArray)

  Gets pollutant values for a specified subcatchment.
- int DLLEXPORT swmm_getGagePrecip (int index, double ∗∗GageArray)

  Get precipitation rates for a gage.
- int DLLEXPORT swmm_getNodeStats (int index, SM_NodeStats ∗nodeStats)

  Get a node statistics.
- int DLLEXPORT swmm_getNodeTotalInflow (int index, double ∗value)

  Get the cumulative inflow for a node.
- int DLLEXPORT swmm_getStorageStats (int index, SM_StorageStats ∗storageStats)

  Get a storage statistics.
- int DLLEXPORT swmm_getOutfallStats (int index, SM_OutfallStats ∗outfallStats)

  Get outfall statistics.
- void DLLEXPORT swmm_freeOutfallStats (SM_OutfallStats ∗outfallStats)

  Free outfall statistics structure.
- int DLLEXPORT swmm_getLinkStats (int index, SM_LinkStats ∗linkStats)

  Get link statistics.
- int DLLEXPORT swmm_getPumpStats (int index, SM_PumpStats ∗pumpStats)

  Get pump statistics.
- int DLLEXPORT swmm_getSubcatchStats (int index, SM_SubcatchStats ∗subcatchStats)

  Get subcatchment statistics.
- int DLLEXPORT swmm_getSystemRoutingStats (SM_RoutingTotals ∗routingTot)

Get system routing statistics.
- int DLLEXPORT swmm_getSystemRunoffStats (SM_RunoffTotals ∗runoffTot)

  Get system runoff statistics.
- int DLLEXPORT swmm_setLinkSetting (int index, double setting)

  Set a link setting (pump, orifice, or weir). Setting for an orifice and a weir should be [0, 1]. A setting for a pump can range from [0, inf). However, if a pump is set to 1, it will pump at its maximum curve setting.
- int DLLEXPORT swmm_setNodeInflow (int index, double flowrate)

  Set an inflow rate to a node. The inflow rate is held constant until the caller changes it.
- int DLLEXPORT swmm_setOutfallStage (int index, double stage)

  Set outfall stage.
- int DLLEXPORT swmm_setGagePrecip (int index, double total_precip)

  Set a total precipitation intensity to the gage.
- void DLLEXPORT freeArray (void ∗∗array)

  Helper function to free memory array allocated in SWMM.

12.3.1 Detailed Description

Exportable Functions for Toolkit API.

See also

`http://github.com/openwateranalytics/stormwater-management-model`

toolkitAPI.c

Date

08/30/2016 (First Contribution)

Authors

B. McDonnell (EmNet LLC), OpenWaterAnalytics members: see `AUTHORS`.

12.3.2 Function Documentation

12.3.2.1 freeArray()

```
void DLLEXPORT freeArray (
             void ** array )
```

Helper function to free memory array allocated in SWMM.

Helper function used to free array allocated memory by API.

Definition at line 1567 of file toolkitAPI.c.

12.3.2.2 newDoubleArray()

```
double * newDoubleArray (
             int n )
```

Warning: Caller must free memory allocated by this function.

Definition at line 1558 of file toolkitAPI.c.

12.3.2.3 swmm_getGagePrecip()

```
int DLLEXPORT swmm_getGagePrecip (
             int index,
             double ** GageArray )
```

Get precipitation rates for a gage.

Input: index = Index of desired ID Output: GageArray pointer (three elements) Return: API Error Purpose: Gets the precipitation value in the gage.

Definition at line 1032 of file toolkitAPI.c.

12.3.2.4 swmm_getObjectIndex()

```
int DLLEXPORT swmm_getObjectIndex (
             int type,
             char * id,
             int * errcode )
```

Gets Object ID Index.

Input: type = object type (Based on SM_ObjectType enum) char∗ = ID name Output: errorcode = pointer to error code Return: Object Index Purpose: Gets object id index

Definition at line 292 of file toolkitAPI.c.

12.3.2.5 swmm_getSubcatchPollut()

```
int DLLEXPORT swmm_getSubcatchPollut (
             int index,
             int type,
             double ** PollutArray )
```

Gets pollutant values for a specified subcatchment.

Input: index = Index of desired ID type = Result Type (SM_SubcPollut) Output: PollutArray pointer (pollutant data desired, byref) Return: API Error Purpose: Gets Subcatchment Simulated Pollutant Value at Current Time

Definition at line 978 of file toolkitAPI.c.

12.3.2.6 swmm_setGagePrecip()

```
int DLLEXPORT swmm_setGagePrecip (
            int index,
            double total_precip )
```

Set a total precipitation intensity to the gage.

Input: index = Index of desired ID total_precip = rainfall intensity to be set Return: API Error Purpose: Sets the precipitation in from the external database

Definition at line 1516 of file toolkitAPI.c.

Index