

Welcome to Rapid Automated-Analysis for Developers (RAAD)'s documentation!

Contents:

- [Rapid Automated-Analysis for Developers \(RAAD\) Framework Sample](#)
 - [Summary](#)
 - [Citing](#)
 - [Developer Information](#)
 - [Organization](#)
- [Related Repositories](#)
- [License](#)
- [Abstract](#)
- [Rapid Automation and Analysis for Developers \(RAAD\) using Telemetry 2.0 User Guide](#)
 - [Introduction](#)
- [RAAD Server with Ubuntu 18+ x64_86 \(Linux\)](#)
 - [Server Platform Recommendation](#)
 - [Training](#)
 - [Memory Installation and Population](#)
 - [Updating the BIOS and Intel Optane Persistent Memory Firmware](#)
 - [How To enter BIOS Recovery Mode](#)
 - [BIOS and Firmware Update Procedure](#)
 - [Compatible Operating Systems for Intel® Optane™ Persistent Memory](#)
 - [Platform Details](#)
 - [How to optimize Optane DIMM usage for DRAM Caching](#)
 - [Technical Details](#)
 - [Ubuntu 18.1x+ LTS x86_64 Instructions](#)
 - [References](#)

- [Documentation Sphinx Generator for Linux](#)
 - [Installation](#)
 - [Quickstart](#)
 - [Python 3.8](#)
 - [Instruction Commands](#)
- [Operating System, Virtual Machine, Container Setup Options](#)
 - [Option 1 Ubuntu 22.04 LTS](#)
 - [Option 2 Ubuntu 22.04 LTS + Docker](#)
 - [Option 3 Create your own image of Ubuntu 22.04 LTS](#)
 - [Option 4 Use a partial make image of Ubuntu 22.04 LTS](#)
 - [Option 5 Intel Pre-made image of Ubuntu 22.04 LTS](#)
 - [Option 6 Windows 10 x86_64 \(Not Recommended\)](#)
 - [Windows 10 x64](#)
- [RAAD Executable Installer for Windows 10 x86_64](#)
 - [Instructions for the installation wizard:](#)
 - [Information for Users of RAAD](#)
 - [Show all docker containers running](#)
 - [Run specific command inside docker container without login into a container](#)
 - [Run a docker image](#)
 - [Log in to specific docker container](#)
 - [Show container logs for docker container](#)
 - [Build and run container on port 8080](#)
 - [Reset docker container and its image](#)
 - [To run container that exited](#)
 - [Reset/truncate docker container logs](#)
 - [## Remove all unused images](#)
- [Anaconda & Jetbrains Environment](#)
 - [Linux Ubuntu Mate x64](#)
 - [Windows 10 x64](#)
- [Telemetry Focused Summary](#)
 - [Mechanisms Architected for Success](#)
 - [Specifications Supported](#)
 - [Telemetry Key Command Information](#)
 - [Summary of Specification Divergence](#)
 - [Type Meta Expectation](#)
 - [Generalized Cases](#)

- [Customer/Developer Data Object Addition Process for Code](#)
- [Access Telemetry using Ubuntu \(Linux\) NVMe-CLI](#)
 - [Summary](#)
 - [Environment](#)
 - [Step-by-step guide](#)
- [Output](#)
- [Troubleshooting](#)
- [NVMe CLI Usage Commands on Linux Debug](#)
 - [Generic Commands](#)
 - [Intel Specific](#)
 - [References](#)
- [Time Series](#)
 - [List of Rapid Learning based on Publications](#)
 - [Videos](#)
 - [Researchers](#)
 - [Research Direction](#)
- [General API Background and Guidelines](#)
 - [Summary](#)
- [Getting Started with RAAD Tools](#)
 - [Instructions for the GUI:](#)
- [RAAD Extended Workloads](#)
 - [Step-by-step guide](#)
- [Phases of Fault Analysis](#)
 - [Phases of Analysis](#)
- [Data Control Assistance for Machine Programming](#)
 - [Related File Overview](#)
- [Definitions for C/C++ Code](#)
- [Guidelines](#)
 - [Data Control Template and Comments](#)
 - [Template Guidelines](#)
- [Accelerating Code Velocity to Integration](#)
 - [Actions Required](#)
- [Detect Data Leaks \(Example\)](#)
- [Mentoring](#)
 - [Guide to Action Required \(ARs\)](#)
 - [Research and Development Guide](#)
- [Patent Construction](#)

- [Features](#)
- [General Guidelines](#)

Indices and tables

- [Index](#)
- [Module Index](#)
- [Search Page](#)

src.software.userConfigurationProfile.main()

class

src.software.userConfigurationProfile.userConfigurationProfile(*debug*)

Bases: **object**

addApplication()

The method to capture user information for a new profile. Returns:
None

class applicationIdentifier

Bases: **object**

applicationList = ['access', 'cloud', 'raad']

configFileName = 'config.ini'

configFolderName = '.raadProfile/'

debug = *True*

getApplicationDefaults(*select=None*)

Gather the default values for given applications. Args:

select: the applicationIdentifier string variable to look at.

Returns: default return values for the current application version.

`getApplicationMeta()`

Gathers applications meta data. Args:

self: Container with internal items.

Returns: String list all application data.

*static getApplications(*self*)*

Gathers the list of short names of applications. Args:

self: Container with internal items.

Returns: String list of application short names.

`getDefaultAccessCLI()`

Function to gather the default access application os system environment. Returns: The path the default paths of given access applications.

`getDefaultHomeSave()`

Determines the current user home configuration directory. Args:

self: Container with internal items.

Returns: home directory configuration save location in current operating system.

*static getInputCommandLine(*self*, *prompt*='Enter information', *default*=None, *sizeMax*=256)*

Method to accept user input and use it in the creation of a configuration file. Args:

self: Container with internal items. prompt: User display information for the command prompt. default: default value in the case of user pressing 'Enter'. sizeMax: Total string size

Returns: The user returned input based on valid inputs of limited size.

getProfileLocation()

Determines the current user home configuration directory. Args:

self: Container with internal items.

Returns: home directory configuration save location in current operating system.

getUserHome()

Determines the current user home directory. Args:

self: Container with internal items.

Returns: home directory location in current operating system.

getUserMeta()

Determines if the input is not none then updates the field. Args:

None

Returns:

identity: unique identification number for a given application.
name: String name of the application. mode: Operation mode of the given application. keyLoc: Location on the user system of the application. encryptionStatus: Flag to determine if the user is currently using encryption on the telemetry data. workingDir: The preferred working location for saving and managing telemetry data.

modifyProfile(*changeSection=None*, *changeVar=None*, *changeValue=None*)

Modifies a given section variables and value in a rprogile. Args:

changeSection: Section in the ini file. changeVar: Option variable name. changeValue: Value to change for a given

profile.

Returns:

`newUser()`

The method to capture user information for a new profile. Returns:
None

`newUserGUI(user_identity, user_name, user_mode, user_keyLoc,
user_encryptionStatus, user_workingDir)`

`printProfile()`

Prints the existing content items in a given profile flushed to the
filesystem. Returns: None

`profileExists()`

Function to determine if a profile already exists. Returns: Returns
the boolean value of existence.

`readProfile()`

Gathers information from existing profile and populates
configuration. Returns: configuration ini and user information read.

`readProfileAsDictionary()`

For a given, read profile into dictionary format. Returns: dictionary
constructed from the internal meta.

`searchProfile(changeSection=None, changeVar=None, raw=False)`

Modifies a given section variables and value in a rprogile. Args:

`changeSection`: Section in the ini file. `changeVar`: Option
variable name. `raw`: All the '%' interpolations are expanded in
the return values, unless the raw argument is true.

Values for interpolation keys are looked up in the same
manner as the option.

Returns: Boolean if found, value, option, section, and configuration object.

`setApplication(select='raad', identity=None, major=None, minor=None, name=None, location=None, mode=None, url=None)`

Determines if the input is not none then updates the field. Args:

select: the applicationIdentifier string variable to look at.
identity: unique identification number for a given application.
major: Major version section to represent a unique organization of app-data. minor: Minor version section to represent an extension of the organization of app-data. name: String name of the application. location: Location on the user system of the application. mode: Operation mode of the given application. url: Application address for accessing data.

Returns: Total count of updated items.

`setUser(identity=None, name=None, mode=None, keyLoc=None, encryptionStatus=None, workingDir=None)`

Determines if the input is not none then updates the field. Args:

identity: unique identification number for a given application. name: String name of the application. mode: Operation mode of the given application. keyLoc: Location on the user system of the application. encryptionStatus: Flag to determine if the user is currently using encryption on the telemetry data. workingDir: The preferred working location for saving and managing telemetry data.

Returns: Total count of updated items.

`class userIdentifier`

Bases: `object`

`validateInput(inputConsole=None, default=False, sizeMax=256)`

Method to accept user input and use it in the creation of a configuration file. Args:

self: Container with internal items. inputConsole: Input to validate. default: default value in the case of user pressing 'Enter'. sizeMax: Total string size

Returns: The user returned input based on valid inputs of limited size.

writeMetaProfile()

Function to create the meta data for a user given the existance of a profile. Returns: None

class src.software.guiOneShot.GUIOneShot(debug=False)

Bases: **object**

Function to define API to run all webAPI sections in one window

ARMAModelPredict()

Returns:

None

ARMAModelPredictConfig()

Returns:

None

AxonDownload()

Downloads content from axon with the configured ID Returns:

None

AxonDownloadConfig()

Configures fields and config file for downloading from the AXON database Returns:

None

AxonUpload()

Upload content file to AXON database Returns:

None

AxonUploadConfig()

DataCollect()

DataCollectConfig()

Spoawns window to change configurations for Data Table method

Returns:

DataTable()

Splits data from .ini files into csv files Returns:

None

DataTableConfig()

Spoawns window to change configurations for Data Table method

Returns:

DefragObjectDecode()

Returns:

None

DefragObjectDecodeConfig()

Returns:

None

static DisplayDebugWindow(*operation, dbugMsg*)

static ExecutePhase(*phase*)

ExecuteTasks(*collect_values*)

Execute tasks with given configurations for Returns:

static FindExistingTelemetryINI()

Asssumes that a time-series_<date>.ini file exists in data/output, or
data/ Returns:

any occurrence of a telemetry config file

GenericObjectDecode()

Returns:

None

GenericObjectDecodeConfig()

Returns:

None

GetConfig()

Look for the location of the config file and config the configparser
memeber variable Returns:

None

GetContentReport()

GetDataFile(section, tag)

GetReport()

OneShotExecuteAPI(ExecuteConfig=None)

OneShotLayout()

Layout for OneShotAPI Returns:

PysimpleGUI layout

OneShotTaskConfig(collect_values)

Configures what tasks are executed by the system Args:

collect_values: list of inputs the user put into the OneShot Window

Returns:

None

RNNModelPredict()

Returns:

None

RNNModelPredictConfig()

Returns:

None

SetPaths()

Set path variables that are used by the OneShotAPI Returns:

None

Window()

baseFilePath = *None*

collectTelemetry = *None*

configLocation = *None*

contentFile = *None*

dataFile = *None*

dataFileName = *None*

debug = *False*

displayReport = *None*

fwParsersLocation = *None*

gatherTelemetryData(*collect_values*)

Near Replication of GUI.gatherTelemetryData() above. This executes the same series of tasks:

1. Query and aggregate telemetry from the chosen SSD
2. Format data into .ini and .txt files
3. Package remaining data files into a compressed format and clean up the directory

Args: collect_values (dict) containing specific configurations for gathering telemetry data

Returns:

dataFileName: datafile valid: boolean which equals true if telemetry is valid

inputLocation = *None*

logoLocation = *None*

outputLocation = *None*

reportDictionary = *None*

reportFlatDictionary = *None*

timeStamp = *None*

validateDataCollectConfig(*dictElem, case*)

writeOneShotProfile(*section, fieldName, value*)

zipFile = *None*

zipName = *None*

src.software.threadModuleAPI.API(*options=None*)

API for the default application in the graphical interface. Args:

options: Commandline inputs.

Returns:

```
class  
src.software.threadModuleAPI.MassiveParallelismSingleFunctionManyParameters(debug: bool = False, functionName=None, fParameters: Optional[List[dict]] = None, workers: Optional[int] = None, timeOut: int = 86400, inOrder: bool = True, runSequential: bool = False)
```

Bases: `object`

`execute()`

`getExceptionInfo()`

`getExecutionTime()`

`getResults()`

`setFunctionName(functionName)`

`setParametersList(fParameters: List[dict])`

Sets parameter list from dictionary parameter context. Args:

`fParameters: list of dictionaries of parameters`

Returns: `None`

Example `kwargsList_input = [{'inputINI': dataFileNameA,`

`... 'debug': debug, 'inParallel': inParallelA, 'requiredList':
requiredListA},`

`...`

`{'inputINI': dataFileNameZ,
... 'debug': debugZ, 'inParallel': inParallelZ,
'requiredList': requiredListZ}]`

```
class
```

```
src.software.threadModuleAPI.MassiveParallelismmanyFunctionManyParameters(debug: bool = False, functionName_fParameters=None, workers:
```

*Optional[int] = None, timeOut: int = 86400, inOrder: bool = True,
runSequential: bool = False)*

Bases: **object**

`execute()`

class src.software.threadModuleAPI.MultiThreadFL

Bases: **object**

Multi-thread a function with multible items.

*threadLoop(items=None, start=None, end=None, num_splits=None,
functionProcess=None)*

*static threadProcessLoop(items=None, start=None, end=None,
threadFunction=None)*

*static threadSplitProcessing(items=None, num_splits=None,
threadProcessFunction=None)*

`src.software.threadModuleAPI.getAvailableCPUCount()`

Number of available virtual or physical CPUs on this system, i.e.
user/real as output by time(1) when called with an optimally scaling
userspace-only program

`src.software.threadModuleAPI.main()`

`src.software.utilsCommon.API(options=None)`

API for the default application in the graphical interface. Args:

options: Commandline inputs.

Returns:

*class src.software.utilsCommon.DictionaryFlatten(inDictionary=None,
debug=False, separator='.', prefix='')*

Bases: `object`

static anyObjectToDictionary(inDictionary)

debug: *bool* = *False*

fdSize: *int* = 0

getFlatDictionary()

getFlattenDictionary(*dd*, separator='.', prefix=')

getOriginalData()

getSize()

getSuperDictionary(*inDictionary=None*, separator='.', prefix=')

originalData = *None*

src.software.utilsCommon.DictionaryLower(*dictionaryForward: dict*)

Function to convert dictionaries with string entries to upper case. Args:

dictionaryForward: Simple 1-dimensional dictionaries.

Returns: dictionary forward in lower case key/values.

src.software.utilsCommon.DictionaryPrune(*queryDict: dict*)

Assistive function to traverse a dictionary recursively then remove data any list to a set entries to comprehend uniqueness. Each dictionary entry contains the reduced size amount from the reference queryDict. Args:

queryDict: Dictionary composed to fundamental types: dict, list, str, int, float, etc.

Returns: Requested dictionary annotations of reduced entries and total entries reduced.

src.software.utilsCommon.DictionaryReduceAnnotator(*queryDict: dict*)

Assistive function to traverse a dictionary recursively then reduce any list to a set entries to comprehend uniqueness. Each dictionary entry contains the reduced size amount from the reference queryDict. Args:

queryDict: Dictionary composed to fundamental types: dict, list, str, int, float, etc.

Returns: Requested dictionary annotations of reduced entries and total entries reduced.

`src.software.utilsCommon.DictionaryReverse(dictionaryForward: dict)`

Function to reverse key with value such that indexes can be used to lookup a key. Args:

dictionaryForward: Simple 1-dimensional dictionaries.

Returns: *dictionaryForward* with Key and Values reversed.

`src.software.utilsCommon.DictionarySetReduce(selfDict: Optional[dict] = None)`

Assistive function to traverse a dictionary recursively then reduce any list to a set removing duplicate entries to comprehend uniqueness. Args:

selfDict: Dictionary composed to fundamental types: dict, list, str, int, float, etc.

Returns: Requested dictionary with lists reduced to sets then converted back to lists.

`src.software.utilsCommon.DictionaryUpper(dictionaryForward: dict)`

Function to convert dictionaries with string entries to upper case. Args:

dictionaryForward: Simple 1-dimensional dictionaries.

Returns: dictionary forward in upper case key/values.

```
class src.software.utilsCommon.FunctionScheduleTimer(debug: bool = False)
```

Bases: **object**

Class to create a function self timer or period query context.

```
calculateDrift(functionCall=None)
```

```
doExamplePeriod(functionCall=None, iterations: int = 1)
```

```
doExamplePeriodExponential(functionCall=None, iterations: int = 1, sleepTime: int = 1)
```

```
doExampleWaitLoop(functionCall=None, iterations: int = 1, allowSleep: bool = False)
```

```
getPeriod(nSeconds: Optional[int] = None)
```

```
static getPeriodExponentialBackoff(baseTime: int = 1, exponentialValue: int = 2, selectedIndex: int = 1)
```

```
getUpdate(functionCall=None)
```

```
isPeriodReady()
```

```
performSleep()
```

```
setup(functionExecutionTime: int = 1)
```

```
src.software.utilsCommon.SimularityScoreCalculate(field_instanceName: str = "", itemOfInterest: str = "")
```

Function to compute the distance simularity score between two candidates. Args:

field_instanceName: Candidate itemOfInterest: query

Returns: Float score [0.0 not similar, 1.0 exact match]

`src.software.utilsCommon.checkPythonVersion(expect_major=3,
expect_minor=9)`

Checking Python version.

To upgrade do the following with Anaconda:

```
conda update conda
conda install python=3.9
conda install anaconda-client
conda update anaconda
conda install -c anaconda python
```

Args:

`expect_major`: python major value. `expect_minor`: python minor value.

Returns: bool if version matches candidates.

`src.software.utilsCommon.checkSum(fileDir=None, chunkSize=8192)`

Get md5 checksum of a file. Chunk size is how much of the file to read at a time. Args:

`fileDir`: File directories. `chunkSize`: Chunk size of the files.

Returns: MD5 hash.

`src.software.utilsCommon.cleanAndRecreatePath(locationOutput:
Optional[str] = None)`

`src.software.utilsCommon.cleanFileName(fileName='defaultName')`

Processes a string name into a filename removing invalid tokens. Args:

`fileName`: string name

Returns: clean name

`src.software.utilsCommon.createFilePath(requestPath)`

Generate file path with existance okay. Args:

`requestPath`: path to request creation.

Returns: Void

src.software.utilsCommon.findAll(*fileType=None*,
directoryTreeRootNode=None, *debug=False*, *verbose=False*, *doIt=False*,
excludeFolderList=None, *excludeFileList=None*)

Find all files of a type given a directory. Args:

fileType: file extension to look for. *directoryTreeRootNode*: filesystem main root node
debug: debug mode for adding functionality. *verbose*: Add more information to debug. *doIt*: Add to systempath. *excludeFolderList*: Folders to not look in.
excludeFileList: files to not look in.

Returns: fileTypeTree, directoryTree

src.software.utilsCommon.flattenList(*inList=None*)

Reduces a list of lists to a single 1-dimension list. Args:

inList: candidate list

Returns: 1-D list

src.software.utilsCommon.generateString(*characterSet=None*,
fileStringNameSize=None)

Generate random string set when not set by user. Args:

characterSet: Character set to be used. *fileStringNameSize*: File string name size maxima.

Returns: Token generated.

src.software.utilsCommon.getBytesSize(*bytesIn=0*, *suffix='B'*)

Scale bytes to its proper format e.g:

1253656 => '1.20MB' 1253656678 => '1.17GB'

src.software.utilsCommon.getDateTimeString()

`src.software.utilsCommon.getFileFormats()`

Dictionary contains file types as keys and lists of their corresponding file formats
Returns: Expected classes of files.

`src.software.utilsCommon.getFileNameUTCTime()`

Get timestamp for a file name Returns: string with UTC time

`src.software.utilsCommon.getPathToRootCount(selectPath: Optional[str] = None)`

`src.software.utilsCommon.getProgramName(fileObj)`

Pass a file object then extract the file name. Args:

fileObj: file object

Returns: file name.

`src.software.utilsCommon.getScriptName(programName)`

Splits a given string suspected to be a file name and returns the prefix with out the file extension. Args:

programName: String file name candidate.

Returns: prefix of file.

`src.software.utilsCommon.getTempFileName(genFile=True)`

Generate file name. Args:

genFile: proceed flag with generation.

Returns: file name token string.

`src.software.utilsCommon.getTempPathAndFileName(extensionName=None, genPath=True, genFile=True)`

Generate path and file name. Args:

extensionName: Extension name desired. genPath: proceed flag with generation. genFile: proceed flag with generation.

Returns: token generation of path and file name token string.

src.software.utilsCommon.getTempPathName(*genPath=True*)

src.software.utilsCommon.getTimeStamp(*inTime=None*)

Time stamp update or set for document. Args:

inTime: date time desired to be set.

Returns: Void

src.software.utilsCommon.main()

src.software.utilsCommon.makeFoldersByFileType(*downloadDirectory=None, fileTypes=None*)

Creates folders for different file types Args:

downloadDirectory: Directory to download or copy files to.

fileTypes: list of filetypes.

Returns: None

src.software.utilsCommon.matchFiles(*ext, directory_files*)

Prints files with extension ext Args:

directory_files: ext:

Returns:

src.software.utilsCommon.moveFile(*moveFiles=None, downloadDirectory=None, fileGroups=None*)

Moves file to its proper folder and delete any duplicates. Args:

`moveFiles`: List of files to move. `downloadDirectory`: Directory to download or copy files to. `fileGroups`: File types to organize by.

Returns:

`src.software.utilsCommon.organizeAPI(options)`

Args:

`options`: options from main file usage.

Returns: None

`src.software.utilsCommon.organizeDirectory(downloadDirectory=None, fileType=None)`

Download of directory by file types. Args:

`downloadDirectory`: Directory to download or copy files to.

`fileTypes`: list of filetypes.

Returns: None

`src.software.utilsCommon.organize_files_by_extension(ext=None, directory_files=None)`

Organize files by the extension. Args:

`ext`: Extension of the file. `directory_files`: Directory containing the files.

Returns: None

`src.software.utilsCommon.organize_files_by_keyword(keyword=None, directory_files=None)`

Args:

`keyword`: keyword string to use in organizing. `directory_files`: Directory containing the files.

Returns:

`src.software.utilsCommon.organize_files_by_letter(first_letter=None, directory_files=None)`

Organize files by letter. Args:

first_letter: The letter order of the candidate. directory_files:
Directory containing the files.

Returns: None

`src.software.utilsCommon.organize_folders_by_keyword(keyword=None, directory_files=None)`

Args:

keyword: keyword string to use in organizing. directory_files:
Directory containing the files.

Returns:

`src.software.utilsCommon.organize_folders_by_letter(first_letter=None, directory_files=None)`

Args:

first_letter: The letter order of the candidate. directory_files:
Directory containing the files.

Returns:

`src.software.utilsCommon.organizeby(FILE_FORMATS: list[str] = None)`

List of file formats to organize by. Args:

FILE_FORMATS: list of lists related to file format.

Returns: None

`src.software.utilsCommon.readFiles(directory_files)`

Args:

directory_files:

Returns:

```
src.software.utilsCommon.strip_StartEnd(stringInput='', prefix='',  
suffix='')
```

```
src.software.utilsCommon.strip_end(stringInput='', suffix='')
```

```
src.software.utilsCommon.strip_start(stringInput='', prefix='')
```

```
src.software.utilsCommon.tryFile(path=None, fileName=None,  
walkUpLimit=4)
```

Helper function to walk up a file path to the destination folder. Args:

path: directory to start with. *fileName*: file to look for in directory
walkUpLimit: levels allowed to walk up in directory
chain

Returns: Full path of the directory

```
src.software.utilsCommon.tryFileDetect(cPath=None, fileName=None)
```

```
src.software.utilsCommon.tryFolder(path=None, walkUpLimit=4)
```

Helper function to walk up a file path to the destination folder. Args:

path: directory name *walkUpLimit*: levels allowed to walk up in
directory chain

Returns: Full path of the directory

```
src.software.utilsCommon.tryFolderDetect(cPath: Optional[str] = None)
```

class src.software.estimationROI.FaultCostModelForReturnOnInvestment

Bases: **object**

```
Cost_Estimation_Model_Bounds(totalFaultCount: int, boundType: str  
= 'upper', phaseOrClassOrModel: str = 'estimationClass')
```

Defintions:

upper is the computer science usage or Big-Oh 'O' or upper bound or worst case. normal is the computer science usage of Theta ' Θ ' of the average case, which is not defined in this usage. lower is the computer science usage or Omega ' Ω ' or lower bound or best case. product phase is the stage in the product life cycle as defined in NASA_Decoder_COL. estimation class is the average, or standard deviation selected in NASA_Decoder_ROW.

Args:

totalFaultCount: boundType: phaseOrClassOrModel:

Returns:

Cost_Estimation_Model_Normalized(*totalFaultCount*)

LCCPS_CostMean(*phaseOrClassOrModel*: str = 'estimationClass')

LCCPS_CostTotal()

LCFPS_Percentage()

static getMinMaxInColOrRow(inputMatrix=None, colOrRow: str = 'row', minOrMax: str = 'max')

class src.software.estimationROI.NASA_Data

Bases: **object**

getDecoderCol()

getDecoderRow()

getLCCPS()

getLCFPS()

class src.software.estimationROI.NPSG_Data

Bases: `object`

class `src.software.estimationROI.NPSG_ProgramData(programName: str, faultCount: int, discoverTime: int, rootCauseTime: int, solutionTime: int, verificationTime: int)`

Bases: `object`

`setAll(programName: Optional[str] = None, faultCount: Optional[int] = None, discoverTime: Optional[int] = None, rootCauseTime: Optional[int] = None, solutionTime: Optional[int] = None, verificationTime: Optional[int] = None)`

`src.software.estimationROI.main()`

`src.software.estimationROI.main_estimator(options, args)`

`src.software.guiTests.API(options=None)`

API for the default application in the graphical interface. Args:

options: Commandline inputs.

Returns:

class `src.software.guiTests.AxonRegressionTests(methodName='runTest')`

Bases: `unittest.case.TestCase`

Regression testing for functions uploading and downloading data to and from AXON

`classmethod addClassCleanup(function, /, *args, **kwargs)`

Same as `addCleanup`, except the cleanup items are called even if `setUpClass` fails (unlike `tearDownClass`).

`addCleanup(function, /, *args, **kwargs)`

Add a function, with arguments, to be called when the test is completed. Functions added are called on a LIFO basis and are

called after tearDown on test failure or success.

Cleanup items are called even if setUp fails (unlike tearDown).

`addTypeEqualityFunc(typeobj, function)`

Add a type specific assertEquals style function to compare a type.

This method is for use by TestCase subclasses that need to register their own type equality functions to provide nicer error messages.

Args:

typeobj: The data type to call this function on when both values are of the same type in assertEquals().

function: The callable taking two arguments and an optional msg= argument that raises self.failureException with a useful error message when the two arguments are not equal.

`assertAlmostEqual(first, second, places=None, msg=None, delta=None)`

Fail if the two objects are unequal as determined by their difference rounded to the given number of decimal places (default 7) and comparing to zero, or by comparing that the difference between the two objects is more than the given delta.

Note that decimal places (from zero) are usually not the same as significant digits (measured from the most significant digit).

If the two objects compare equal then they will automatically compare almost equal.

`assertAlmostEquals(**kwargs)`

`assertCountEqual(first, second, msg=None)`

Asserts that two iterables have the same elements, the same number of times, without regard to order.

```
self.assertEqual(Counter(list(first)),  
                 Counter(list(second)))
```

Example:

- [0, 1, 1] and [1, 0, 1] compare equal.
- [0, 0, 1] and [0, 1] compare unequal.

`assertDictContainsSubset(subset, dictionary, msg=None)`

Checks whether dictionary is a superset of subset.

`assertDictEqual(d1, d2, msg=None)`

`assertEqual(first, second, msg=None)`

Fail if the two objects are unequal as determined by the '`==`' operator.

`assertEquals(**kwargs)`

`assertFalse(expr, msg=None)`

Check that the expression is false.

`assertGreater(a, b, msg=None)`

Just like `self.assertTrue(a > b)`, but with a nicer default message.

`assertGreaterEqual(a, b, msg=None)`

Just like `self.assertTrue(a >= b)`, but with a nicer default message.

`assertIn(member, container, msg=None)`

Just like `self.assertTrue(a in b)`, but with a nicer default message.

`assertIs(expr1, expr2, msg=None)`

Just like `self.assertTrue(a is b)`, but with a nicer default message.

`assertIsInstance(obj, cls, msg=None)`

Same as `self.assertTrue(isinstance(obj, cls))`, with a nicer default message.

`assertIsNone(obj, msg=None)`

Same as `self.assertTrue(obj is None)`, with a nicer default message.

`assertIsNot(expr1, expr2, msg=None)`

Just like `self.assertTrue(a is not b)`, but with a nicer default message.

`assertIsNotNone(obj, msg=None)`

Included for symmetry with `assertIsNone`.

`assertLess(a, b, msg=None)`

Just like `self.assertTrue(a < b)`, but with a nicer default message.

`assertLessEqual(a, b, msg=None)`

Just like `self.assertTrue(a <= b)`, but with a nicer default message.

`assertListEqual(list1, list2, msg=None)`

A list-specific equality assertion.

Args:

`list1`: The first list to compare. `list2`: The second list to compare.

`msg`: Optional message to use on failure instead of a list of

differences.

`assertLogs(logger=None, level=None)`

Fail unless a log message of level `level` or higher is emitted on `logger_name` or its children. If omitted, `level` defaults to INFO and `logger` defaults to the root logger.

This method must be used as a context manager, and will yield a recording object with two attributes: `output` and `records`. At the end of the context manager, the `output` attribute will be a list of the matching formatted log messages and the `records` attribute will be a list of the corresponding LogRecord objects.

Example:

```
with self.assertLogs('foo', level='INFO') as cm:
    logging.getLogger('foo').info('first message')
    logging.getLogger('foo.bar').error('second message')
self.assertEqual(cm.output, ['INFO:foo:first message',
                            'ERROR:foo.bar:second
message'])
```

`assertMultiLineEqual(first, second, msg=None)`

Assert that two multi-line strings are equal.

`assertNotAlmostEqual(first, second, places=None, msg=None, delta=None)`

Fail if the two objects are equal as determined by their difference rounded to the given number of decimal places (default 7) and comparing to zero, or by comparing that the difference between the two objects is less than the given delta.

Note that decimal places (from zero) are usually not the same as significant digits (measured from the most significant digit).

Objects that are equal automatically fail.

`assertNotAlmostEquals(**kwargs)`

`assertNotEqual(first, second, msg=None)`

Fail if the two objects are equal as determined by the `'!='` operator.

`assertNotEquals(**kwargs)`

`assertNotIn(member, container, msg=None)`

Just like `self.assertTrue(a not in b)`, but with a nicer default message.

`assertNotIsInstance(obj, cls, msg=None)`

Included for symmetry with `assertIsInstance`.

`assertNotRegex(text, unexpected_regex, msg=None)`

Fail the test if the text matches the regular expression.

```
assertNotRegexpMatches(**kwargs)
```

```
assertRaises(expected_exception, *args, **kwargs)
```

Fail unless an exception of class `expected_exception` is raised by the callable when invoked with specified positional and keyword arguments. If a different type of exception is raised, it will not be caught, and the test case will be deemed to have suffered an error, exactly as for an unexpected exception.

If called with the callable and arguments omitted, will return a context object used like this:

```
with self.assertRaises(SomeException):
    do_something()
```

An optional keyword argument 'msg' can be provided when `assertRaises` is used as a context object.

The context manager keeps a reference to the exception as the 'exception' attribute. This allows you to inspect the exception after the assertion:

```
with self.assertRaises(SomeException) as cm:
    do_something()
the_exception = cm.exception
self.assertEqual(the_exception.error_code, 3)
```

```
assertRaisesRegex(expected_exception, expected_regex, *args,
                  **kwargs)
```

Asserts that the message in a raised exception matches a regex.

Args:

`expected_exception`: Exception class expected to be raised.

`expected_regex`: Regex (re.Pattern object or string) expected

to be found in error message.

`args`: Function to be called and extra positional args.

`kwargs`: Extra kwargs.

`msg`: Optional message used in case of failure.

Can only be used

when assertRaisesRegex is used as a context manager.

`assertRaisesRegexp(**kwargs)`

`assertRegex(text, expected_regex, msg=None)`

Fail the test unless the text matches the regular expression.

`assertRegexpMatches(**kwargs)`

`assertSequenceEqual(seq1, seq2, msg=None, seq_type=None)`

An equality assertion for ordered sequences (like lists and tuples).

For the purposes of this function, a valid ordered sequence type is one which can be indexed, has a length, and has an equality operator.

Args:

`seq1`: The first sequence to compare. `seq2`: The second sequence to compare. `seq_type`: The expected datatype of the sequences, or `None` if no

datatype should be enforced.

`msg`: Optional message to use on failure instead of a list of differences.

`assertSetEqual(set1, set2, msg=None)`

A set-specific equality assertion.

Args:

`set1`: The first set to compare. `set2`: The second set to compare.

`msg`: Optional message to use on failure instead of a list of

differences.

`assertSetEqual` uses ducktyping to support different types of sets, and is optimized for sets specifically (parameters must support a difference method).

`assertTrue(expr, msg=None)`

Check that the expression is true.

`assertTupleEqual(tuple1, tuple2, msg=None)`

A tuple-specific equality assertion.

Args:

`tuple1`: The first tuple to compare. `tuple2`: The second tuple to compare. `msg`: Optional message to use on failure instead of a list of differences.

`assertWarns(expected_warning, *args, **kwargs)`

Fail unless a warning of class `warnClass` is triggered by the callable when invoked with specified positional and keyword arguments. If a different type of warning is triggered, it will not be handled: depending on the other warning filtering rules in effect, it might be silenced, printed out, or raised as an exception.

If called with the callable and arguments omitted, will return a context object used like this:

```
with self.assertWarns(SomeWarning):
    do_something()
```

An optional keyword argument '`msg`' can be provided when `assertWarns` is used as a context object.

The context manager keeps a reference to the first matching warning as the '`warning`' attribute; similarly, the '`filename`' and '`lineno`' attributes give you information about the line of Python code from which the warning was triggered. This allows you to inspect the warning after the assertion:

```
with self.assertWarns(SomeWarning) as cm:  
    do_something()  
the_warning = cm.warning  
self.assertEqual(the_warning.some_attribute, 147)
```

`assertWarnsRegex(expected_warning, expected_regex, *args, **kwargs)`

Asserts that the message in a triggered warning matches a regexp.
Basic functioning is similar to `assertWarns()` with the addition that
only warnings whose messages also match the regular expression
are considered successful matches.

Args:

`expected_warning`: Warning class expected to be triggered.
`expected_regex`: Regex (re.Pattern object or string) expected
to be found in error message.

`args`: Function to be called and extra positional args. `kwargs`:
Extra kwargs. `msg`: Optional message used in case of failure.
Can only be used

when `assertWarnsRegex` is used as a context manager.

`assert_(*kwargs)`

`countTestCases()`

`debug()`

Run the test without collecting errors in a `TestResult`

`defaultTestResult()`

`classmethod doClassCleanups()`

Execute all class cleanup functions. Normally called for you after
`tearDownClass`.

`doCleanups()`

Execute all cleanup functions. Normally called for you after `tearDown`.

`fail(msg=None)`

Fail immediately, with the given message.

`failIf(**kwargs)`

`failIfAlmostEqual(**kwargs)`

`failIfEqual(**kwargs)`

`failUnless(**kwargs)`

`failUnlessAlmostEqual(**kwargs)`

`failUnlessEqual(**kwargs)`

`failUnlessRaises(**kwargs)`

`failureException`

alias of `AssertionError`

`id()`

`longMessage = True`

`maxDiff = 640`

`run(result=None)`

`setUp()`

Function Type: Helper

`classmethod setUpClass()`

Hook method for setting up class fixture before running tests in the class.

`shortDescription()`

Returns a one-line description of the test, or None if no description has been provided.

The default implementation of this method returns the first line of the specified test method's docstring.

`skipTest(reason)`

Skip this test.

`subTest(msg=<object object>, **params)`

Return a context manager that will return the enclosed block of code in a subtest identified by the optional message and keyword parameters. A failure in the subtest marks the test case as failed but resumes execution at the end of the enclosed block, allowing further test code to be executed.

`tearDown()`

Function Type: Helper

`classmethod tearDownClass()`

Hook method for deconstructing the class fixture after running all tests in the class.

`testCorrectUploadID()`

Sends expected file to upload and checks the ID received Returns:

Success: Whether test had the expected output

`testDownloadBadID()`

Prompts AXON to download an ID that does not exist Returns:

Success: Whether test had the expected output

`testDownloadSuccess()`

Prompts AXON to download an ID that does exist Returns:

Success: Whether test had the expected output

`testEmptyUpload()`

Sends empty file to upload Returns:

Success: Whether test had the expected output

`testUploadFail()`

Sends bad file to upload Returns:

Success: Whether test had the expected output

`testUploadSuccess()`

Sends expected file to upload Returns:

Success: Whether test had the expected output

`testUploadSuccessDev()`

Sends expected file to upload in the development space Returns:

Success: Whether test had the expected output

class `src.software.guiTests.TableRegressionTests(methodName='runTest')`

Bases: `unittest.case.TestCase`

Regression testing for functions creating the telemetry data table

class`method addClassCleanup(function, /, *args, **kwargs)`

Same as `addCleanup`, except the cleanup items are called even if `setUpClass` fails (unlike `tearDownClass`).

`addCleanup(function, /, *args, **kwargs)`

Add a function, with arguments, to be called when the test is completed. Functions added are called on a LIFO basis and are called after `tearDown` on test failure or success.

Cleanup items are called even if `setUp` fails (unlike `tearDown`).

`addTypeEqualityFunc(typeobj, function)`

Add a type specific assertEqual style function to compare a type.

This method is for use by TestCase subclasses that need to register their own type equality functions to provide nicer error messages.

Args:

`typeobj`: The data type to call this function on when both values are of the same type in `assertEqual()`.

`function`: The callable taking two arguments and an optional `msg=` argument that raises `self.failureException` with a useful error message when the two arguments are not equal.

`assertAlmostEqual(first, second, places=None, msg=None, delta=None)`

Fail if the two objects are unequal as determined by their difference rounded to the given number of decimal places (default 7) and comparing to zero, or by comparing that the difference between the two objects is more than the given delta.

Note that decimal places (from zero) are usually not the same as significant digits (measured from the most significant digit).

If the two objects compare equal then they will automatically compare almost equal.

`assertAlmostEquals(**kwargs)`

`assertCountEqual(first, second, msg=None)`

Asserts that two iterables have the same elements, the same number of times, without regard to order.

```
self.assertEqual(Counter(list(first)),  
                Counter(list(second)))
```

Example:

- [0, 1, 1] and [1, 0, 1] compare equal.
- [0, 0, 1] and [0, 1] compare unequal.

`assertDictContainsSubset(subset, dictionary, msg=None)`

Checks whether dictionary is a superset of subset.

`assertDictEqual(d1, d2, msg=None)`

`assertEqual(first, second, msg=None)`

Fail if the two objects are unequal as determined by the '==' operator.

`assertEquals(**kwargs)`

`assertFalse(expr, msg=None)`

Check that the expression is false.

`assertGetDataAsArray(testInput, expectedResults, chosenKey='all', oneShot=False)`

Assert checking for outputs that are received from the getDataAsArray function Args:

`testInput`: input DataDict that will be passed to getDataAsArray
`*expectedResults`: Resulting array that is formed chosenKey:
`oneShot`:

Returns:

`assertGreater(a, b, msg=None)`

Just like self.assertTrue(a > b), but with a nicer default message.

`assertGreaterEqual(a, b, msg=None)`

Just like self.assertTrue(a >= b), but with a nicer default message.

`assertIn(member, container, msg=None)`

Just like self.assertTrue(a in b), but with a nicer default message.

`assertIs(expr1, expr2, msg=None)`

Just like `self.assertTrue(a is b)`, but with a nicer default message.

`assertIsInstance(obj, cls, msg=None)`

Same as `self.assertTrue(isinstance(obj, cls))`, with a nicer default message.

`assertIsNone(obj, msg=None)`

Same as `self.assertTrue(obj is None)`, with a nicer default message.

`assert IsNot(expr1, expr2, msg=None)`

Just like `self.assertTrue(a is not b)`, but with a nicer default message.

`assertIsNotNone(obj, msg=None)`

Included for symmetry with `assertIsNone`.

`assertLess(a, b, msg=None)`

Just like `self.assertTrue(a < b)`, but with a nicer default message.

`assertLessEqual(a, b, msg=None)`

Just like `self.assertTrue(a <= b)`, but with a nicer default message.

`assertListEqual(list1, list2, msg=None)`

A list-specific equality assertion.

Args:

`list1`: The first list to compare. `list2`: The second list to compare.

`msg`: Optional message to use on failure instead of a list of

differences.

`assertLogs(logger=None, level=None)`

Fail unless a log message of level `level` or higher is emitted on `logger_name` or its children. If omitted, `level` defaults to INFO and `logger` defaults to the root logger.

This method must be used as a context manager, and will yield a recording object with two attributes: *output* and *records*. At the end of the context manager, the *output* attribute will be a list of the matching formatted log messages and the *records* attribute will be a list of the corresponding LogRecord objects.

Example:

```
with self.assertLogs('foo', level='INFO') as cm:  
    logging.getLogger('foo').info('first message')  
    logging.getLogger('foo.bar').error('second message')  
self.assertEqual(cm.output, ['INFO:foo:first message',  
                           'ERROR:foo.bar:second  
message'])
```

`assertMultiLineEqual(first, second, msg=None)`

Assert that two multi-line strings are equal.

`assertNotAlmostEqual(first, second, places=None, msg=None,
delta=None)`

Fail if the two objects are equal as determined by their difference rounded to the given number of decimal places (default 7) and comparing to zero, or by comparing that the difference between the two objects is less than the given delta.

Note that decimal places (from zero) are usually not the same as significant digits (measured from the most significant digit).

Objects that are equal automatically fail.

`assertNotAlmostEquals(**kwargs)`

`assertNotEqual(first, second, msg=None)`

Fail if the two objects are equal as determined by the `'!='` operator.

`assertNotEquals(**kwargs)`

`assertNotIn(member, container, msg=None)`

Just like `self.assertTrue(a not in b)`, but with a nicer default message.

`assertNotIsInstance(obj, cls, msg=None)`

Included for symmetry with `assertIsInstance`.

`assertNotRegex(text, unexpected_regex, msg=None)`

Fail the test if the text matches the regular expression.

`assertNotRegexpMatches(**kwargs)`

`assertRaises(expected_exception, *args, **kwargs)`

Fail unless an exception of class `expected_exception` is raised by the callable when invoked with specified positional and keyword arguments. If a different type of exception is raised, it will not be caught, and the test case will be deemed to have suffered an error, exactly as for an unexpected exception.

If called with the callable and arguments omitted, will return a context object used like this:

```
with self.assertRaises(SomeException):
    do_something()
```

An optional keyword argument 'msg' can be provided when `assertRaises` is used as a context object.

The context manager keeps a reference to the exception as the 'exception' attribute. This allows you to inspect the exception after the assertion:

```
with self.assertRaises(SomeException) as cm:
    do_something()
the_exception = cm.exception
self.assertEqual(the_exception.error_code, 3)
```

`assertRaisesRegex(expected_exception, expected_regex, *args, **kwargs)`

Asserts that the message in a raised exception matches a regex.

Args:

expected_exception: Exception class expected to be raised.
expected_regex: Regex (re.Pattern object or string) expected

to be found in error message.

args: Function to be called and extra positional args. kwargs:
Extra kwargs. msg: Optional message used in case of failure.
Can only be used

when assertRaisesRegex is used as a context manager.

`assertRaisesRegexp(**kwargs)`

`assertRegex(text, expected_regex, msg=None)`

Fail the test unless the text matches the regular expression.

`assertRegexpMatches(**kwargs)`

`assertSequenceEqual(seq1, seq2, msg=None, seq_type=None)`

An equality assertion for ordered sequences (like lists and tuples).

For the purposes of this function, a valid ordered sequence type is one which can be indexed, has a length, and has an equality operator.

Args:

seq1: The first sequence to compare. seq2: The second sequence to compare. seq_type: The expected datatype of the sequences, or None if no

datatype should be enforced.

msg: Optional message to use on failure instead of a list of differences.

`assertSetEqual(set1, set2, msg=None)`

A set-specific equality assertion.

Args:

set1: The first set to compare. set2: The second set to compare.
msg: Optional message to use on failure instead of a list of
differences.

assertSetEqual uses ducktyping to support different types of sets,
and is optimized for sets specifically (parameters must support a
difference method).

assertTrue(*expr*, *msg=None*)

Check that the expression is true.

assertTupleEqual(*tuple1*, *tuple2*, *msg=None*)

A tuple-specific equality assertion.

Args:

tuple1: The first tuple to compare. tuple2: The second tuple to
compare. msg: Optional message to use on failure instead of a
list of
differences.

assertWarns(*expected_warning*, **args*, *kwargs*)**

Fail unless a warning of class warnClass is triggered by the callable
when invoked with specified positional and keyword arguments. If a
different type of warning is triggered, it will not be handled:
depending on the other warning filtering rules in effect, it might be
silenced, printed out, or raised as an exception.

If called with the callable and arguments omitted, will return a
context object used like this:

```
with self.assertWarns(SomeWarning):  
    do_something()
```

An optional keyword argument 'msg' can be provided when
assertWarns is used as a context object.

The context manager keeps a reference to the first matching warning as the 'warning' attribute; similarly, the 'filename' and 'lineno' attributes give you information about the line of Python code from which the warning was triggered. This allows you to inspect the warning after the assertion:

```
with self.assertWarns(SomeWarning) as cm:  
    do_something()  
the_warning = cm.warning  
self.assertEqual(the_warning.some_attribute, 147)
```

`assertWarnsRegex(expected_warning, expected_regex, *args, **kwargs)`

Asserts that the message in a triggered warning matches a regexp. Basic functioning is similar to `assertWarns()` with the addition that only warnings whose messages also match the regular expression are considered successful matches.

Args:

`expected_warning`: Warning class expected to be triggered.
`expected_regex`: Regex (re.Pattern object or string) expected to be found in error message.

`args`: Function to be called and extra positional args. `kwargs`: Extra kwargs. `msg`: Optional message used in case of failure. Can only be used

when `assertWarnsRegex` is used as a context manager.

`assert_(**kwargs)`

`countTestCases()`

`debug()`

Run the test without collecting errors in a `TestResult`

`defaultTestResult()`

*class*method `doClassCleanups()`

Execute all class cleanup functions. Normally called for you after `tearDownClass`.

`doCleanups()`

Execute all cleanup functions. Normally called for you after `tearDown`.

`fail(msg=None)`

Fail immediately, with the given message.

`failIf(**kwargs)`

`failIfAlmostEqual(**kwargs)`

`failIfEqual(**kwargs)`

`failUnless(**kwargs)`

`failUnlessAlmostEqual(**kwargs)`

`failUnlessEqual(**kwargs)`

`failUnlessRaises(**kwargs)`

`failureException`

alias of `AssertionError`

`id()`

`longMessage = True`

`maxDiff = 640`

`run(result=None)`

`setUp()`

Function Type: Helper

classmethod `setUpClass()`

Hook method for setting up class fixture before running tests in the class.

`shortDescription()`

Returns a one-line description of the test, or `None` if no description has been provided.

The default implementation of this method returns the first line of the specified test method's docstring.

`skipTest(reason)`

Skip this test.

`subTest(msg=<object object>, **params)`

Return a context manager that will return the enclosed block of code in a subtest identified by the optional message and keyword parameters. A failure in the subtest marks the test case as failed but resumes execution at the end of the enclosed block, allowing further test code to be executed.

`tearDown()`

Function Type: Helper

classmethod `tearDownClass()`

Hook method for deconstructing the class fixture after running all tests in the class.

`testGetDataDict()`

`testNilContainer()`

`testOneShotContainer()`

`testPopupTable()`

`testSimpleContainer()`

```
testSpecificContainer()  
src.software.guiTests.main()  
src.software.guiTests.suite()  
src.software.guiDeveloper.API(options=None)
```

API for the default application in the graphical interface. Args:

options: Commandline inputs.

Returns:

```
class src.software.guiDeveloper.GUIDeveloper(name=None, label=None,  
layout=None, form=None, windowActive=None, button=None,  
values=None, debug=False)
```

Bases: `object`

```
static GUIDownload(axonID, outputDirectory, mode='test',  
analysisReport=None)
```

```
GUIUpload(contentFile=None, mode='test', analysisReport=None)
```

`add_buttons()`

Submit and cancel button for a layout. Returns: None

```
add_selectDropDown(selectionList=None, size=(20, 3))
```

Drop down menu example Args:

selectionList: List of selection string options. size: Box size information.

Returns: None

```
add_selectFolder(text='Choose Content Folder',  
setupBox=_

---


```

', size=(35, 1))

Selection to choose a local folder for a given operation. Args:

text: Text box to display to user. setupBox: Horizontal line breaking text. size: Textbox size information.

Returns: None

add_typeTextBox(text='Please provide feedback based on experience', size=(35, 3))

Add a text input box to the application. Args:

text: Text box to let user be aware of the input intent. size: Text box size

Returns: None

add_windowLabel(text='Please enter your System Information')

Diag. box information for the layout. Args:

text: Text box to display in layout.

Returns:

static check_flag(flag)

function for checking the value of a yes/no drop-down and returning the boolean value associated with the answer

Args:

flag: string representation of Yes/No option

Returns:

8 booleanFlag: Boolean value for selected option ("Yes" = True, "No" = False)

static create_new_graph_window_ARMA(values, mep, currentObject, pdf=None)

function for generating a new graph window using the matplotlib functionality of PySimpleGUI

Args:

values: List of values collected from window mep:
MediaErrorPredictor instance currentObject: String for the name of the current object
pdf: Instance of the PDF file descriptor where the graphs will be stored. If pdf==None, the plots will be directly displayed to the screen

Returns:

create_new_graph_window_RNN(values, rnn, currentObject, primaryVarLabelsRNN, secondaryVarLabelsRNN)

function for generating a new graph window using the matplotlib functionality of PySimpleGUI

Args:

values: List of values collected from window rnn:
mediaPredictionRNN instance currentObject: String for the name of the current object (ex. uid-6)
primaryVarLabelsRNN: List of strings identifiers for the variable fields used as inputs in the RNN
secondaryVarLabelsRNN: List of strings identifiers for the variable fields to be plotted

Returns:

static create_new_graph_window_defrag(values, dhg, dhFile, setPointLabelsDict, primaryVarLabelsDefragHistory, secondaryVarLabelsDefragHistory)

function for generating a new graph window using the matplotlib functionality of PySimpleGUI

Args:

values: List of values collected from window dhg:
DefragHistoryGrapher instance dhFile: DefragConfig instance
setPointLabelsDict: List of window element names that collect the setpoint values primaryVarLabelsDefragHistory: String identifier for the fields in the GUI that contains the variable names to be plotted in the main axis of the graph secondaryVarLabelsDefragHistory: String identifier for the fields in the GUI that contains the variable names to be plotted in the secondary axis of the graph

Returns:

static create_new_graph_window_generic_object(values, vts, objectFile, currentObject, primaryVarLabelsGenericObject, secondaryVarLabelsGenericObject)

function for generating a new graph window using the matplotlib functionality of PySimpleGUI

Args:

values: List of values collected from window vts: visualizeTS instance objectFile: ObjectConfig instance currentObject: String for the name of the current object (ex. uid-6) primaryVarLabelsGenericObject: String identifier for the fields in the GUI that contains the variable names to be plotted in the main axis of the graph secondaryVarLabelsGenericObject: String identifier for the fields in the GUI that contains the variable names to be plotted in the secondary axis of the graph

Returns:

static display_dir_ARMA(window, objectFile, currentObject)

function for updating the values contained in the drop down menus for tracking variables

Args:

window: window instance objectFile: ObjectConfig instance currentObject: String for the name of the current object

Returns:

*static display_dir_RNN(window, objectFile, currentObject,
primaryVarLabelsRNN, secondaryVarLabelsRNN)*

function for updating the values contained in the drop down menus
for primary and secondary variables

Args:

window: window instance objectFile: ObjectConfig instance
currentObject: String for the name of the current object
primaryVarLabelsRNN: List of strings identifiers for the
variable fields used as inputs in the RNN
secondaryVarLabelsRNN: List of strings identifiers for the
variable fields to be plotted

Returns:

*static display_dir_defrag(window, dirPath, mode,
primaryVarLabelsDefragHistory, secondaryVarLabelsDefragHistory)*

function for updating the values contained in the drop down menus
for primary and secondary variables

Args:

window: window instance dirPath: String for the path to the
configuration file mode: Integer for the mode of operation
(1=ADP, 2=CDR) primaryVarLabelsDefragHistory: String
identifier for the fields in the GUI that contains the variable
names to be plotted in the main axis of the graph
secondaryVarLabelsDefragHistory: String identifier for the fields
in the GUI that contains the variable names to be plotted in the
secondary axis of the graph

Returns:

dhg: DefragHistoryGrapher instance dhFile: DefragConfig
instance

static display_dir_generic_object(window, objectFile, currentObject, primaryVarLabelsGenericObject, secondaryVarLabelsGenericObject)

function for updating the values contained in the drop down menus for primary and secondary variables

Args:

window: window instance
objectFile: ObjectConfig instance
currentObject: String for the name of the current object (ex. uid-6)
primaryVarLabelsGenericObject: String identifier for the fields in the GUI that contains the variable names to be plotted in the main axis of the graph
secondaryVarLabelsGenericObject: String identifier for the fields in the GUI that contains the variable names to be plotted in the secondary axis of the graph

Returns:

display_windowInput()

Display the layout composed. Returns: None

downloadAXONData()

Spawns new window to get information and then performs a download from the AXON database Returns:

execute_nlog_predictor(values)

function for executing the standard NLOG predictor. Access to the terminal output is required to see the results

Args:

values: List of values collected from window

Returns:

formMetaData(axonInt, contentFile)

Creates metadata file for AXON upload Args: axonInt: contentFile:

Returns: Path to MetaData file used for AXON upload

`gatherTelemetryData(collect_values)`

Reaction to pressing the Gather Data button on the gui. That button press kicks off a series of three major actions:

1. Query and aggregate telemetry from the chosen SSD
2. Format data into .ini and .txt files
3. Package remaining data files into a compressed format and clean up the directory

Args:

`collect_values`: Array containing the user inputs from the graphical interface

`static getAXONIDs()`

Gather a dictionary for details of previous AXON uploads Returns:

`axonList`: list of python dictionaries that contain relevant details of previous axon uploads

The dictionary follow this format: [{"descriptor": "value", ...}, ...]

`static getDataAsArray(DataDic, chosenKey='all', oneShot=False)`

`static getDataDic(dataFile)`

`getMenu()`

Gets the default menu selection. Returns: None

`getProgramLabel()`

Gets the default label for the program. Returns: None

`get_all()`

Method to access self saved data. Args:

`self`: All data related to internal tracked information.

Returns: All data related to internal tracked information.

`static get_tableOptions()`

Method to get the python GUI table Options. Args: None Returns: all table options

`get_windowInputs()`

Returns current window information. Args:

self: Internal value used in button and value tracking.

Returns: Button status and value passed in.

`static hostMetaData()`

Get MetaData that has to do with who is retrieving the information
Returns:

dict[str: str] hostMeta: Contains the fields for host-specific data

`loadAndProbeDrive(collect_values)`

`make_table(num_rows=1, num_cols=1)`

Generation of tables. Args:

`num_rows`: Number of rows to generate. `num_cols`: Number of columns to generate

Returns: None

`static number(max_val=4)`

Generation of random numbers. Args:

`max_val`: Total values to generate

Returns: Random number vector.

`static populate_object_ARMA(window, dirPath)`

function for populating the ObjectConfig instance and updating the drop down menu for the objects available in the configuration file

Args:

window: window instance dirPath: String for the path to the configuration file

Returns:

mep: MediaErrorPredictor instance objectFile: ObjectConfig instance

static populate_object_RNN(window, dirPath)

function for populating the ObjectConfig instance and updating the drop down menu for the objects available in the configuration file

Args:

window: window instance dirPath: String for the path to the configuration file

Returns:

rnn: mediaPredictionRNN instance objectFile: ObjectConfig instance objectList: list of object names (ex. ThermalSensor)

static populate_object_generic_object(window, dirPath)

function for populating the ObjectConfig instance and updating the drop down menu for the objects available in the configuration file

Args:

window: window instance dirPath: String for the path to the configuration file

Returns:

vts: visualizeTS instance objectFile: ObjectConfig instance objectNamesDict: Dictionary of object identifiers (ex. uid-6) to object names (ex. ThermalSensor)

popupTable(dataArray, programLabel, returnLayout=False)

static popupTableLayout(dataArray, headings)

print_all()

Method to print out the current content in the class. Args:

self: Internal variables

Returns: None

`print_packageLibraryVersion()`

Print information related to the library currently in use. Args:

self: Used to access debug flag.

Returns: Python simple GUI information.

`read_window()`

Reads the active window. Returns: None

`refreshMetaTable()`

refresh data in Meta Data Table in Graphical Interface

`sendAXONDownloadCommand()`

Returns:

`setAppearance(color='LightBlue')`

Method to set the color and feel of the layout. Args:

color: Selected color from list.

Returns: None

`spawnAxonDownloadWindow()`

`trackAXONUpload(axonID=0, metaData=None, axonFile='raadProfile/axonProfile.ini')`

Function to pull in/create a hidden file in the current working directory and pull information about previously uploaded AXON records.

Creates a file called .raadProfile/axonProfile.ini and stores metadata for each upload based on the metadata uploaded to AXON

Returns:

None

updateTimeStamp()

webAPI()

static word(wordWidth=10)

Random word creator Args:

wordWidth: Word length.

Returns: String generated word.

src.software.guiDeveloper.main()

src.software.guiLayouts.API(*options=None*)

API for the default application in the graphical interface. Args:

options: Commandline inputs.

Returns:

class src.software.guiLayouts.GUILayouts(*debug=False*)

Bases: **object**

List of example tabs for the baseline application.

static ARMAPredictionGraph(returnLayout=False)

function for generating the graphical interface window for graphing media error predictions

Args:

returnLayout: Boolean flag to access the graphing layout produced in the function

Returns:

layout: None or the layout of the window

static GUIDownload(axonID, outputDirectory, mode='test', analysisReport=None)

GUIUpload(contentFile=None, mode='test', analysisReport=None)

static RNNPredictorGraph(returnLayout=False)

function for generating the graphical interface window for graphing an RNN time series predictor object

Args:

returnLayout: Boolean flag to access the graphing layout produced in the function

Returns:

layout: None or the layout of the window

static collect(returnLayout=False, charWidth=20, charHeight=1, debug=False)

Layout construction for an independent user profile. Args:

returnLayout: Flag to return the flag or execute in independent mode. charWidth: Width of the text box. charHeight: Height of a given display box. debug: debug print flag.

Returns: layout or input box information.

debug = False

static debugComments(returnLayout=False, displayInput=None, charWidth=93, charHeight=26, visible=True)

Layout construction for an independent user profile. Args:

returnLayout: Flag to return the flag or execute in independent mode. displayInput: Strings variables to Display. charWidth: Width of the text box. charHeight: Height of a given display box. visible: Visibility of the given object.

Returns: layout or input box information.

static defragHistoryGraph(returnLayout=False)

function for generating the graphical interface window for graphing a defragHistory object

Args:

returnLayout: Boolean flag to access the graphing layout produced in the function

Returns:

layout: None or the layout of the window

download(returnLayout=False, charWidth=25, charHeight=1)

Layout construction for downloading AXON data Args:

returnLayout: Flag to return the flag or execute in independent mode. charWidth: Width of the text box. charHeight: Height of a given display box.

Returns: layout or input box information.

formMetaData(axonInt, contentFile)

Creates metadata file for AXON upload Args: axonInt: contentFile:

Returns: Path to MetaData file used for AXON upload

static hostMetaData()

Get MetaData that has to do with who is retrieving the information
Returns:

dict[str: str] hostMeta: Contains the fields for host-specific data

static neuralNetClassify(returnLayout=False)

Layout construction for an independent user profile. Args:

returnLayout: Flag to return the flag or execute in independent mode.

Returns: layout or input box information.

static objectTimeSeriesVisualizer(returnLayout=False)

static profileApplication(returnLayout=False, charWidth=25, charHeight=1, identity=None, major=None, minor=None, name=None, location=None, mode=None, url=None)

Layout construction for an independent user profile. Args:

returnLayout: Flag to return the flag or execute in independent mode. charWidth: Width of the text box. charHeight: Height of a given display box. identity: unique identification number for a given application. major: Major version section to represent a unique organization of app-data. minor: Minor version section to represent an extension of the organization of app-data. name: String name of the application. location: Location on the user system of the application. mode: Operation mode of the given application. url: Application address for accessing data.

Returns: layout or input box information.

static profileUser(returnLayout=False, charWidth=25, charHeight=1, identity=None, name=None, mode=None, keyLoc=None, encryptionStatus=None, workingDir=None)

Layout construction for an independent user profile. Args:

returnLayout: Flag to return the flag or execute in independent mode. charWidth: Width of the text box. charHeight: Height of a given display box. identity: unique identification number for a given application. name: String name of the application. mode: Operation mode of the given application. keyLoc:

Location on the user system of the application.

encryptionStatus: Flag to determine if the user is currently using encryption on the telemetry data. workingDir: The preferred working location for saving and managing telemetry data.

Returns: layout or input box information.

`tabAPI()`

`trackAXONUpload(axonID=0, metaData=None,
axonFile='.raadProfile/axonProfile.ini')`

Function to pull in/create a hidden file in the current working directory and pull information about previously uploaded AXON records.

Creates a file called .raadProfile/axonProfile.ini and stores metadata for each upload based on the metadata uploaded to AXON

Returns:

None

`upload(returnLayout=False, charWidth=25, charHeight=1)`

Layout construction for an independent user profile. Args:

returnLayout: Flag to return the flag or execute in independent mode. charWidth: Width of the text box. charHeight: Height of a given display box.

Returns: layout or input box information.

`static userFeedback(returnLayout=False, charWidth=25,
charHeight=1)`

Layout construction for an independent user profile. Args:

returnLayout: Flag to return the flag or execute in independent mode. charWidth: Width of the text box. charHeight: Height of

a given display box.

Returns: layout or input box information.

class src.software.guiLayouts.dataTablePopulate(*dataIn=None*,
returnLayout=False, *charWidth=25*, *charHeight=1*, *debug=False*)

Bases: **object**

charHeight = 1

charWidth = 25

dataIn = None

debug = False

static getDataAsArray(DataDic)

static getDataDic(dataFile)

main()

make_table(num_rows=1, num_cols=1)

Generation of tables. Args:

num_rows: Number of rows to generate. *num_cols*: Number of columns to generate

Returns: None

static number(max_val=4)

Generation of random numbers. Args:

max_val: Total values to generate

Returns: Random number vector.

popupTable(dataArrayLocal, programLabelLocal)

static popupTableLayout(*dataArrayLocal*, *headings*)

returnLayout = *False*

static word(*wordWidth*=10)

Random word creator Args:

wordWidth: Word length.

Returns: String generated word.

src.software.guiLayouts.main()

class src.software.guiCommon.DefragConfig(*configFilePath*, *mode*)

Bases: **object**

dhFilePath = *None*

readConfigContent(*debug*=*False*)

function for reading the configuration file into a dictionary and populating the DefragConfig instance

Args:

debug: Boolean flag to activate debug statements

Returns:

dhg: DefragHistoryGrapher instance

secondaryVars = []

setPoints = []

trackingVars = []

class src.software.guiCommon.GenericObjectGraph

Bases: **object**

Generating the graphical interface window for graphing a generic object

`create_new_graph_window(valuesVar, vtsVar, objectFileVar, currentObjectVar)`

function for generating a new graph window using the matplotlib functionality of PySimpleGUI

Args:

valuesVar: List of values collected from window
vtsVar: visualizeTS instance
objectFileVar: ObjectConfig instance
currentObjectVar: String for the name of the current object (ex. uid-6)

Returns:

`display_dir(windowVar, objectFileVar, currentObjectVar)`

function for updating the values contained in the drop down menus for primary and secondary variables

Args:

windowVar: window instance
objectFileVar: ObjectConfig instance
currentObjectVar: String for the name of the current object (ex. uid-6)

Returns:

`static is_data_selected_from_fields(listVar)`

function for checking if data has been selected for a list

Args:

listVar: list of data values

Returns:

`static populate_object(windowVar, dirPathVar)`

function for populating the ObjectConfig instance and updating the drop down menu for the objects available in the configuration file

Args:

windowVar: window instance dirPathVar: String for the path to the configuration file

Returns:

vtsVarLocal: visualizeTS instance objectFileVarLocal: ObjectConfig instance objectNamesDict: Dictionary of object identifiers (ex. uid-6) to object names (ex. ThermalSensor)

primaryVarLabels = ['-PFVAR-', '-PSVAR-', '-PTVAR-']

secondaryVarLabels = ['-SFVAR-', '-SSVAR-', '-STVAR-']

class src.software.guiCommon.ObjectConfigARMA(*configFilePath*)

Bases: **object**

objectFilePath = *None*

objectIDs = []

readConfigContent(*debug=False*)

function for reading the configuration file into a dictionary and populating the ObjectConfig instance

Args:

debug: Boolean flag to activate debug statements

Returns:

mep: MediaErrorPredictor instance

trackingVars = []

class src.software.guiCommon.ObjectConfigGenericObject(*configFilePath*, *debug=False*)

Bases: **object**

GetObjectIDDecode()

objectFilePath = *None*

objectIDDecode = *None*

objectIDs = []

readConfigContent(*debug=None*)

function for reading the configuration file into a dictionary and populating the ObjectConfig instance

Returns:

vts: visualizeTS instance

vizDict = *None*

vts = *None*

class src.software.guiCommon.ObjectConfigRNN(*configFilePath*)

Bases: **object**

objectFilePath = *None*

objectIDs = []

readConfigContent(*debug=False*)

function for reading the configuration file into a dictionary and populating the ObjectConfig instance

Args:

debug: Boolean flag to activate debug statements

Returns:

rnn: mediaPredictionRNN instance

class src.software.guiCommon.collectGUI

Bases: **object**

getDictSet()

getVector(*key*)

Returns the row of data for a given key. Args:

key:

Returns:

`isPossible(key=None, value=None)`

Method to determine if we can set a value field to a given value from possible existing values
Args:

key: Key within the vector. value: Value with associated with the key.

Returns: Returns if the assignment for a given key value pair is possible.

`keyExists(key=None)`

Determines if the keys exist in the current vector. Args:

key: key string to search for in the array.

Returns: If the key is found and the index found.

`valueSet(key=None, value=None)`

Method to search for a key value pair then return the vector location.
Args:

key: Key to look for. value: Value to set for the given key.

Returns: The vector location of the row for the given key.

`class src.software.guiCommon.dictElement(nKey=None, nValue=None, nType=None, nPossible=None, nDefault=None, nPrompt=None, nTooltip=None)`

Bases: `object`

`getDefault()`

`getInputs()`

getKey()
getPossible()
getPrompt()
getTooltip()
getType()
getValue()
setValue(*value*)

src.software.autoModuleAPI.API(options=None)

API for the default application in the graphical interface. Args:

options: Commandline inputs.

Returns:

class src.software.autoModuleAPI.autoModuleMeta

Bases: **object**

src.software.autoModuleAPI.generateARMA(inputINI=None, subSequenceLength=1, matrixProfileFlag=None, debug=False, inParallel=True, requiredList=None, outFolder=None, timeOut=180, max_workers=None)

Generate all ARMA models for data. Args:

inParallel: requiredList: List of strings for the names of objects to be processed if the useRequiredList flag is set.

If None, the default list will be used. Indicates whether the objects to be processed should be limited to the ones contained in the requiredList.

inputINI: *.ini with preprocessed time series. subSequenceLength: [1 to len(time_series)/3] sequence length for matrix profile.
matrixProfileFlag: Boolean flag that enables matrix profile data generation. debug: [True, False] developer debug statements.
outFolder: Output directory. timeOut: Execution timeout.
max_workers: total number of threads.

Returns: list of PDFs of ARMA models

src.software.autoModuleAPI.generateAXONDownload(*axonID=None, downloadDirectory=None, downloadMode='test', debug=False*)

User download content data file gathered to the AXON database. Args:

axonID: identifier to download downloadDirectory: directory to save content downloadMode: search database debug: developer debug logs.

Returns: status information and meta data from download.

src.software.autoModuleAPI.generateAXONUpload(*contentFile=None, uploadMode='test', userName='lab', debug=False*)

User upload content data file gathered to the AXON database. Args:

contentFile: Upload info filename for JSON creation. uploadMode: 'production', 'test', 'development', 'base'. userName: user name with ability to upload. debug: Developer debug statement.

Returns: status information and meta data from upload.

src.software.autoModuleAPI.generateAllContent(*debug=True, logoFileName='software/Intel_IntelligentSystems.png', inputFolder='data/inputSeries', autoParseFolder='Auto-Parse/decode/ADP_UV', dataCntrlFolder='Auto-Parse/datacontrol', nlogFolder='software/parse/nlogParser', outputFolder='data/output', credentialsFile='.raadProfile/credentials.conf', reportName='RAD_Report', username='lab', password='lab', axonMode='test', subSequenceLength=1, matrixProfileFlag=False, inputWidth=70, labelWidth=20, shift=2*,

```
hiddenLayers=128, batchSize=32, maxEpochs=2096,  
embeddedEncodingFlag=False, categoricalEncodingFlag=False,  
inParallel=False, inParallelLocal=False, inOrder=True,  
requiredList=None, skipPhases=None)
```

Generation of content in one simple API. Args:

nlogFolder: Nlog meta folder debug: Developer debug flag.
logoFileName: Product logo image. inputFolder: Input directory
with telemetry binaries. autoParseFolder: auto parser folders.
dataCntrlFolder: data control folder. outputFolder: output folder for
generated files. credentialsFile: credential cache file with name and
password. reportName: Output report name. username: User name
for RAAD. password: Password name for RAAD. axonMode:
Axon mode of operation. subSequenceLength: Sequence length for
matrix profile. matrixProfileFlag: Flag to perform matrix profile.
inputWidth: Input data width for AI NN. labelWidth: Label data
width for AI NN. shift: AI shift variable. hiddenLayers: AI NN
hidden layers. batchSize: AI batch size. maxEpochs: AI Epochs.
embeddedEncodingFlag: AI embedded encoding for meta data.
categoricalEncodingFlag: AI catagorical encoding flag for meta
data. inParallel: Parallel sub-functions. inParallelLocal: Local
parallel functions. inOrder: Flag to process out of order or in order.
Be careful with data dependency. requiredList: List of strings for
the names of objects to be processed indicating whether the objects
to be processed should be limited to the ones. Contained in the
requiredList. If None, the default list will be used. skipPhases:
Phases in the auto API to skip

Returns: Display report dictionary.

```
src.software.autoModuleAPI.generateBinaryDataSet(driveNumber=0,  
driveName=None, inputFile=None, identifier='Tv2HiTAC', outputDir='.',  
parse=False, volumeLabel='perftest', volumeAllocUnit='4096',  
volumeFS='ntfs', volumeLetter='g', partitionStyle='mbr',  
partitionFlag=True, prepFlag=True, debug=False, performanceTest=True)
```

Preparation of NVMe drive then gathering of data based on
performance or media collection benchmarks. Args:

driveNumber: string representation of int for drive number.
driveName: driveName: Name of device interface to get data from.
inputFile: String for the path to the input file where the workload configuration is stored. identifier: String for the name of the data set that corresponds to the telemetry pull to be executed. outputDir: String for the path to the output directory where the binaries will be stored. parse: Boolean flag to parse the telemetry binaries pulled from the drive. volumeLabel: (Windows) String for the label to be used on the disk volume. volumeAllocUnit: (Windows) String for the volume allocation unit size. volumeFS: (Windows) String for the name of the file system to be used in the disk volume.
volumeLetter: (Windows) String for the letter to be assigned to the disk volume. partitionStyle: (Windows) String for the name of the partition style to be used in the specified disk. partitionFlag: (Windows) Boolean flag to indicate if the program should partition the drive using the given parameters. prepFlag: (Linux) Boolean flag to indicate if the program should prep the drive before loading it. debug: Boolean flag to activate debug statements.
performanceTest: Perform standard performance test or media profile test.

Returns: Success of API 0 is good status.

`src.software.autoModuleAPI.generateDataTables(inputINI=None,
saveCSV=True, outputLocation='.', debug=False)`

Generates data tables from telemetry information Args:

inputINI: input time series file to process. *saveCSV*: Flag to save data as CSV *outputLocation*: output folder save location *debug*: developer debug flag

Returns: uid list, tables, size, and files

`src.software.autoModuleAPI.generateDefragHistory(inputINI=None,
outputFile='telemetryDefault', mode=1, numCores=1,
bandwidthFlag=True, debug=False)`

Generate all relevant graphs associated with the DefragHistory telemetry object Args:

inputINI: relative path to the .ini file where the time series are stored
outputFile: string for the prefix to be used for the PDF file naming mode: Integer value for run mode (1=ADP, 2=CDR)
numCores: Integer for the number of cores from which data was pulled
bandwidthFlag: Boolean flag that indicates if the secondary axis corresponds to bandwidth debug: [True, False] developer debug statements.

Returns: Name of the PDF file where all the graphs are stored

```
src.software.autoModuleAPI.generateHandbook(username='PresidentSkroob@spaceballs.one', password='12345', passwordEncodingObjectSaved=None, defaultPasswordFile='../../raadProfile/credentials.conf', outCacheFile='../../raadProfile/debugHandbookCache.ini', searchString='ASSERT_DE003', similarityScoreThreshold=0.95, debug=False)
```

Wiki page download and save Args:

username: Potential Username password: Potential Password
passwordEncodingObjectSaved: Previously encoded username and password.
defaultPasswordFile: Secret file location with username and password
outCacheFile: Output wiki cache file.
searchString: Debug string to look for.
similarityScoreThreshold: similarity score for looking through debug items.
debug: developer debug flag.

Returns: Password object, hash of password

```
src.software.autoModuleAPI.generateJIRAClusters(debug=True, faultSignature='ASSERT_DE003', outputFolder='data/output', credentialsFile='raadProfile/credentials.conf', logoFileName='software/Intel_IntelligentSystems.png', inputFolder='data/inputSeries', dataCntrlFolder='Auto-Parse/datacontrol', autoParseFolder='Auto-Parse/decode/ADP_UV', nlogFolder='.')
```

```
filePrefix='Tv2HiTAC', fileTokenNamePostfix='TokenData.ini',
fileZipNamePostfix='MetaData.zip',
fileAnalysisLogName='autoModule.log')
```

Generation of content in one simple API. Args:

fileAnalysisLogName: log analysis file fileZipNamePostfix: filename token fileTokenNamePostfix: filename token filePrefix: set Prefix for processing nlogFolder: negative log folder autoParseFolder: Tokenizers folder dataCntrlFolder: data control folder. inputFolder: input directory logoFileName: logo file to be used faultSignature: fault code. debug: Developer debug flag. outputFolder: output folder for generated files. credentialsFile: credential cache file with name and password.

Returns: Display report dictionary.

```
src.software.autoModuleAPI.generateParseDataSet(containingFolder='.',
autoParseFolder='.', outFolder='.', nlogFolder='.', operationMode=3,
autoParseMode=2, debug=False, fileIdentifier='Tv2HiTAC',
timeName=None, fileTokenName='TokenData.ini',
fileMetaName='MetaData.zip', debugFile=None, skipZip: bool = False)
```

Function to parse telemetry data payloads. Args:

debugFile: data dictionary debug file. containingFolder: Folder with the input binary files. autoParseFolder: Folder with the auto parse decoders. outFolder: Folder to target generated files. nlogFolder: Log folder and tokenizers operationMode: 1:CLI, 2: TWIDL, 3: PARSE, 4: IMAS. autoParseMode: 1: bufferDictionary, 2: autoParsers. debug: Debug developer information. fileIdentifier: File prefix. timeName: Time token name. fileTokenName: Token ini name. fileMetaName: Meta Data zip name. skipZip: Parameter to skip zipping content

Returns: Generated information and objects from data processsing.

```
src.software.autoModuleAPI.generateRNNLSTM(inputINI=None,
subSequenceLength=1, matrixProfileFlag=False, inputWidth=70,
```

```
labelWidth=20, shift=2, hiddenLayers=128, batchSize=32,
maxEpochs=2096, embeddedEncodingFlag=False,
categoricalEncodingFlag=False, debug=False, inParallel=True,
requiredList=None, useEntirePayload=False, timeOut=180,
max_workers=None, inOrder: bool = True)
```

Generate all ARMA models for data. Args:

inParallel: Boolean flag to kick off process in parallel requiredList: List of strings for the names of objects to be processed if the useRequiredList flag is set.

If None, the default list will be used. Indicates whether the objects to be processed should be limited to the ones contained in the requiredList.

useEntirePayload: Boolean flag to indicate that the entire telemetry payload should be fed into the RNN model as an input

inputINI: *.ini with preprocessed time series. inputWidth: Integer for the length of the input sequence to be considered for the prediction labelWidth: Integer for the length of the output sequence expected from the prediction shift: Integer for the shift to be considered when sliding the input window hiddenLayers: Integer for the number of neurons contained in each hidden layer subSequenceLength: [1 to len(time_series)/3] sequence length for matrix profile. batchSize: Integer for the size of the batch to be considered when generating data sets to be fed into the RNN model. maxEpochs: Integer for the maximum number of epochs to be considered when training the model matrixProfileFlag: Boolean flag that enables matrix profile data generation.

embeddedEncodingFlag: Boolean flag to apply embedded encoding to time series as the first layer in the RNN model

categoricalEncodingFlag: Boolean flag to apply label encoding to the time series values (usually used for categorical values) debug: developer debug statements. timeOut: Execution timeout.

max_workers: Number of tasks to execute on in parallel. inOrder:

out-of-order or in-order execution. Dependencies on data should set flag to in-order.

Returns: list of PDFs of ARMA models.

```
src.software.autoModuleAPI.generateTimeSeriesGraphs(inputINI=None,  
outputFile='telemetryDefault', subSequenceLength=1,  
matrixProfileFlag=False, inParallel=True, requiredList=None,  
debug=False, timeOut=180)
```

Generate all the graphs for the telemetry objects contained in inputINI
Args:

inParallel: Flag to process all objects in parallel processes.

requiredList: List of strings for the names of objects to be processed if the useRequiredList flag is set.

If None, the default list will be used. Indicates whether the objects to be processed should be limited to the ones contained in the requiredList.

inputINI: relative path to the .ini file where the time series are stored
outputFile: string for the prefix to be used for the PDF file naming
subSequenceLength: Integer for the length of the sliding window for matrix profile (only relevant, if matrix profile flag is set)
matrixProfileFlag: Boolean flag to apply matrix profile to time series before plotting it
debug: [True, False] developer debug statements.
timeOut: Execution timeout.

Returns: List of file names for the PDFs containing the graphs (one PDF for each object, and one graph per field)

```
src.software.autoModuleAPI.main()
```

class

```
src.software.axon.axonProfile.AxonProfile(profile_path='/home/jdtarang/rad/raad/dox', profile_description=None, profileFile=None)
```

Bases: **object**

AddSection(*entry*, *ID*)

Returns:

GetProfile(*profileFile*='.raadProfile/axonProfile.ini')

Get config parser and reads the already-created file if it exists

Returns:

configParser: configParser that can parse the formats from .ini files

SaveProfile(*profileFile*='.raadProfile/axonProfile.ini')

```
class src.software.axon.axonMeta.AXON_Meta(uid=- 1, major=- 1,  
minor=1, time=datetime.datetime(2022, 6, 29, 18, 37, 51, 322522),  
user='DarthVader', debug=True)
```

Bases: **object**

GetVersion()

static RAAD(*self*)

```
RADString(idName='RAD_PF', idVersion='1', pythonVersion='3',  
appType='collector', modeVersion='development')
```

Application content creator for Rapid Automation-Analysis for Developers.

Args:

idName (str): Application identification name. *idVersion* (str): Application identification version. *pythonVersion* (str): Python version. *appType* (str): Application mode type. *modeVersion* (str): Mode version.

Returns:

List[str]: Construction of the string to be uploaded in describing the contents.

static content(*self*, *fileName*=")

```
contentString(content_version='1', fileListName=None,  
source_version='1', collection_type='collector', name_version='1.0',  
python_version='3', error_encountered=")
```

Content description package. Args:

content_version (str): Version of the content
fileListName (List[str]): File list of the content
source_version (str): Version of the content
collection_type (str): Type either collection or decoder.
name_version (str): Name of the content.
python_version (str): Python dependency version.
error_encountered (str): Errors encountered.

Returns:

List[str]: Construction of the string to be uploaded in describing the contents.

```
contents(product='Arbordale Plus',  
sha='b91c544280d55d9e3e2139d7337bd94733442f82f727e0a79ec3295  
ea2a67f78', timeStamp='2020-04-01T15:24:01.339000',  
uuidRandom='326e95920ffa4ff98ee6c8b48ff9d0a3',  
infoSource='Telemetry', collectorTimeStamp='2020-04-01T21:24:20',  
fileName='drive_info.zip', phase='WORKLOAD', host='lm-310-19-h1',  
testName='StressERDTTest_logs', bus='NVMe',  
numberOfSectors=33230736, driveStatus='Healthy',  
serial='PHAB00000000960CGN', wwid=0, BlockDataSize=512,  
MaximumTransferSize=131072, model='INTEL  
SSDPF2KX960G9SSME2', firmware='2DAAZ000', testTag='ADP-XFT-  
QUALITYANDRELIABILITY-2020WW13.7_101345',  
testRunID='test_run_id', pool='RAD_Dev_PF')
```

Creation of the meta data for a zip file to be uploaded. Args:

product (str): Product identifier. *sha* (str): SHA signature generated from file. *timeStamp* (str): Time stamp of event. *uuidRandom* (str): unique signature generator. *infoSource* (str): Source of the information. *collectorTimeStamp* (str): Collection from the time stamp. *fileName* (str): File name to upload. *phase* (str): Phase of the workload generated. *host* (str):

Host name of the system. `testName` (str): Test execution name. `bus` (str): Type of interface used. `numberOfSectors` (str): Total sectors of the device. `driveStatus` (str): Drive state. `serial` (str): serial of the device. `wwid` (str): World wide identifier of creator. `BlockDataSize` (str): Block size used in the device. `MaximumTransferSize` (str): Maximum transfer size supported by the device. `model` (str): Model of the device. `firmware` (str): Firmware on the device. `testTag` (str): Execution generated tag of the device. `testRunID` (str): The run identifier of the execution. `pool` (str): System pool associated.

Returns:

`List[str]`: Construction of the string to be uploaded in describing the contents.

`convertTime(year=1970, month=1, day=1, hour=0, minute=0, second=0, microsecond=0)`

Function to calculate the time from epoch or a given signature. Args:

`@year`: Year of epoch 1970. `@month`: Month of epoch January.
`@day`: Day of epoch 1. `@hour`: Hour of epoch 0. `@minute`: Minute of epoch 0. `@second`: Second of epoch 0.
`@microsecond`: Microsecond of epoch 0.

Return:

UTC time in seconds from epoch.

`driveInfoZip(product='Arbordale Plus', sha='b91c544280d55d9e3e2139d7337bd94733442f82f727e0a79ec3295ea2a67f78', timeStamp='2020-04-01T15:24:01.339000', uuidRandom='326e95920ffa4ff98ee6c8b48ff9d0a3', infoSource='Telemetry', collectorTimeStamp='2020-04-01T21:24:20', fileName='drive_info.zip', phase='WORKLOAD', host='lm-310-19-h1', testName='StressERDTTest_logs', bus='NVMe', numberOfSectors=33230736, driveStatus='Healthy', serial='PHAB0000000960CGN', wwid=0, BlockDataSize=512, MaximumTransferSize=131072, model='INTEL`

SSDPF2KX960G9SSME2', firmware='2DAAZ000', testTag='ADP-XFT-QUALITYANDRELIABILITY-2020WW13.7_101345', testRunID='test_run_id', pool='RAD_Dev_PF')

Creation of the meta data for a zip file to be uploaded. Args:

product (str): Product identifier. sha (str): SHA signature generated from file. timeStamp (str): Time stamp of event. uuidRandom (str): unique signature generator. infoSource (str): Source of the information. collectorTimeStamp (str): Collection from the time stamp. fileName (str): File name to upload. phase (str): Phase of the workload generated. host (str): Host name of the system. testName (str): Test execution name. bus (str): Type of interface used. numberOfWorkers (str): Total workers of the device. driveStatus (str): Drive state. serial (str): serial of the device. wwid (str): World wide identifier of creator. BlockDataSize (str): Block size used in the device. MaximumTransferSize (str): Maximum transfer size supported by the device. model (str): Model of the device. firmware (str): Firmware on the device. testTag (str): Execution generated tag of the device. testRunID (str): The run identifier of the execution. pool (str): System pool associated.

Returns:

List[str]: Construction of the string to be uploaded in describing the contents.

failure(hostName=None, pool='RAD_Dev_PF', version=1, osInfo='linux', axonCLIVersion='', uuid=None)

static host(self)

loadJSON(inJSONFile=None)

static meta(self)

static platform(self)

```
record(hostName='skynetT2020', pool'RAD_Dev_PF', version=1,  
osInfo=None, axonCLIVersion'1.0.17-e48ba68', uuid=None)
```

Record meta creator for the overall instance. Args:

 @*hostName*: Name of the host machine. @*pool*: Name of the machine pool. @*version*: Version of the record creation.
 @*osInfo*: Operating system identification. @*axonCLIVersion*: Version identifier
 @*uuid*: Unique identifier generator for queries.

Returns:

 String array of the record for file writing.

```
saveJSON(saveFile=None, objectJSON=None)
```

static source(self)

static system(self)

static ticket(self)

```
ticketString(hsdes_id=0, hsdes_link'http://link_to_hsd',  
cirs_id'CI1927-3292',  
cirs_link'http://link_to_speed_cirs_facr_ticket', cirs_type'FACR',  
jira_id=0, jira_link'http://link_to_jira', jira_info'world')
```

JIRA bug tracking content. Args:

hsdes_id (int): Identification number. *hsdes_link* (): Web hyperlink to content. *cirs_id* (): Identification number. *cirs_link* (): Web hyperlink to content. *cirs_type* (str): Description type. *jira_id* (str): JIRA identification. *jira_link* (str): Web hyperlink to content. *jira_info* (str): Description information.

Returns:

 List[str]: Construction of the string to be uploaded in describing the contents.

`src.software.axon.axonMeta.hollowShell(options=None)`

`src.software.axon.axonMeta.main()`

`src.software.axon.axonMeta.setup(debug=True)`

`class src.software.axon.axonInterface.Axon_Interface(uid=-1, major=-1, minor=1, time=datetime.datetime(2022, 6, 29, 18, 37, 51, 322760), user='DarthVader', debug=True, mode='base', axonPath=None)`

Bases: `object`

`GetURL()`

`GetVersion()`

`class TestAxonProducer(methodName='runTest')`

Bases: `unittest.case.TestCase`

`classmethod addClassCleanup(function, /, *args, **kwargs)`

Same as `addCleanup`, except the cleanup items are called even if `setUpClass` fails (unlike `tearDownClass`).

`addCleanup(function, /, *args, **kwargs)`

Add a function, with arguments, to be called when the test is completed. Functions added are called on a LIFO basis and are called after `tearDown` on test failure or success.

Cleanup items are called even if `setUp` fails (unlike `tearDown`).

`addTypeEqualityFunc(typeobj, function)`

Add a type specific `assertEqual` style function to compare a type.

This method is for use by `TestCase` subclasses that need to register their own type equality functions to provide nicer error messages.

Args:

`typeobj`: The data type to call this function on when both values
are of the same type in `assertEqual()`.

`function`: The callable taking two arguments and an optional
`msg=` argument that raises `self.failureException` with a
useful error message when the two arguments are not
equal.

`assertAlmostEqual(first, second, places=None, msg=None,
delta=None)`

Fail if the two objects are unequal as determined by their
difference rounded to the given number of decimal places
(default 7) and comparing to zero, or by comparing that the
difference between the two objects is more than the given delta.

Note that decimal places (from zero) are usually not the same as
significant digits (measured from the most significant digit).

If the two objects compare equal then they will automatically
compare almost equal.

`assertAlmostEquals(**kwargs)`

`assertCountEqual(first, second, msg=None)`

Asserts that two iterables have the same elements, the same
number of times, without regard to order.

`self.assertEqual(Counter(list(first)),
Counter(list(second)))`

Example:

- [0, 1, 1] and [1, 0, 1] compare equal.
- [0, 0, 1] and [0, 1] compare unequal.

`assertDictContainsSubset(subset, dictionary, msg=None)`

Checks whether dictionary is a superset of subset.

`assertDictEqual(d1, d2, msg=None)`

`assertEqual(first, second, msg=None)`

Fail if the two objects are unequal as determined by the '`==`' operator.

`assertEquals(**kwargs)`

`assertFalse(expr, msg=None)`

Check that the expression is false.

`assertGreater(a, b, msg=None)`

Just like `self.assertTrue(a > b)`, but with a nicer default message.

`assertGreaterEqual(a, b, msg=None)`

Just like `self.assertTrue(a >= b)`, but with a nicer default message.

`assertIn(member, container, msg=None)`

Just like `self.assertTrue(a in b)`, but with a nicer default message.

`assertIs(expr1, expr2, msg=None)`

Just like `self.assertTrue(a is b)`, but with a nicer default message.

`assertIsInstance(obj, cls, msg=None)`

Same as `self.assertTrue(isinstance(obj, cls))`, with a nicer default message.

`assertIsNone(obj, msg=None)`

Same as `self.assertTrue(obj is None)`, with a nicer default message.

`assert IsNot(expr1, expr2, msg=None)`

Just like `self.assertTrue(a is not b)`, but with a nicer default message.

`assertIsNotNone(obj, msg=None)`

Included for symmetry with `assertIsNone`.

`assertLess(a, b, msg=None)`

Just like `self.assertTrue(a < b)`, but with a nicer default message.

`assertLessEqual(a, b, msg=None)`

Just like `self.assertTrue(a <= b)`, but with a nicer default message.

`assertListEqual(list1, list2, msg=None)`

A list-specific equality assertion.

Args:

`list1`: The first list to compare. `list2`: The second list to compare. `msg`: Optional message to use on failure instead of a list of differences.

`assertLogs(logger=None, level=None)`

Fail unless a log message of level `level` or higher is emitted on `logger_name` or its children. If omitted, `level` defaults to INFO and `logger` defaults to the root logger.

This method must be used as a context manager, and will yield a recording object with two attributes: `output` and `records`. At the end of the context manager, the `output` attribute will be a list of the matching formatted log messages and the `records` attribute will be a list of the corresponding LogRecord objects.

Example:

```
with self.assertLogs('foo', level='INFO') as cm:  
    logging.getLogger('foo').info('first message')  
    logging.getLogger('foo.bar').error('second  
message')
```

```
self.assertEqual(cm.output, ['INFO:foo:first message',
                           'ERROR:foo.bar:second
message'])
```

assertMultiLineEqual(*first*, *second*, *msg=None*)

Assert that two multi-line strings are equal.

assertNotAlmostEqual(*first*, *second*, *places=None*, *msg=None*, *delta=None*)

Fail if the two objects are equal as determined by their difference rounded to the given number of decimal places (default 7) and comparing to zero, or by comparing that the difference between the two objects is less than the given delta.

Note that decimal places (from zero) are usually not the same as significant digits (measured from the most significant digit).

Objects that are equal automatically fail.

assertNotAlmostEquals(***kwargs*)

assertNotEqual(*first*, *second*, *msg=None*)

Fail if the two objects are equal as determined by the ' \neq ' operator.

assertNotEquals(***kwargs*)

assertNotIn(*member*, *container*, *msg=None*)

Just like `self.assertTrue(a not in b)`, but with a nicer default message.

assertNotIsInstance(*obj*, *cls*, *msg=None*)

Included for symmetry with `assertIsInstance`.

assertNotRegex(*text*, *unexpected_regex*, *msg=None*)

Fail the test if the text matches the regular expression.

```
assertNotRegexpMatches(**kwargs)
```

```
assertRaises(expected_exception, *args, **kwargs)
```

Fail unless an exception of class `expected_exception` is raised by the callable when invoked with specified positional and keyword arguments. If a different type of exception is raised, it will not be caught, and the test case will be deemed to have suffered an error, exactly as for an unexpected exception.

If called with the callable and arguments omitted, will return a context object used like this:

```
with self.assertRaises(SomeException):
    do_something()
```

An optional keyword argument 'msg' can be provided when `assertRaises` is used as a context object.

The context manager keeps a reference to the exception as the '`exception`' attribute. This allows you to inspect the exception after the assertion:

```
with self.assertRaises(SomeException) as cm:
    do_something()
the_exception = cm.exception
self.assertEqual(the_exception.error_code, 3)
```

```
assertRaisesRegex(expected_exception, expected_regex, *args,
**kwargs)
```

Asserts that the message in a raised exception matches a regex.

Args:

`expected_exception`: Exception class expected to be raised.

`expected_regex`: Regex (`re.Pattern` object or string) expected

to be found in error message.

`args`: Function to be called and extra positional args. `kwargs`:

Extra kwargs. `msg`: Optional message used in case of failure.

Can only be used

when assertRaisesRegex is used as a context manager.

`assertRaisesRegexp(**kwargs)`

`assertRegex(text, expected_regex, msg=None)`

Fail the test unless the text matches the regular expression.

`assertRegexpMatches(**kwargs)`

`assertSequenceEqual(seq1, seq2, msg=None, seq_type=None)`

An equality assertion for ordered sequences (like lists and tuples).

For the purposes of this function, a valid ordered sequence type is one which can be indexed, has a length, and has an equality operator.

Args:

`seq1`: The first sequence to compare. `seq2`: The second sequence to compare. `seq_type`: The expected datatype of the sequences, or `None` if no

datatype should be enforced.

`msg`: Optional message to use on failure instead of a list of differences.

`assertSetEqual(set1, set2, msg=None)`

A set-specific equality assertion.

Args:

`set1`: The first set to compare. `set2`: The second set to compare. `msg`: Optional message to use on failure instead of a list of

differences.

`assertSetEqual` uses ducktyping to support different types of sets, and is optimized for sets specifically (parameters must support a difference method).

`assertTrue(expr, msg=None)`

Check that the expression is true.

`assertTupleEqual(tuple1, tuple2, msg=None)`

A tuple-specific equality assertion.

Args:

`tuple1`: The first tuple to compare. `tuple2`: The second tuple to compare. `msg`: Optional message to use on failure instead of a list of differences.

`assertWarns(expected_warning, *args, **kwargs)`

Fail unless a warning of class `warnClass` is triggered by the callable when invoked with specified positional and keyword arguments. If a different type of warning is triggered, it will not be handled: depending on the other warning filtering rules in effect, it might be silenced, printed out, or raised as an exception.

If called with the callable and arguments omitted, will return a context object used like this:

```
with self.assertWarns(SomeWarning) :  
    do_something()
```

An optional keyword argument '`msg`' can be provided when `assertWarns` is used as a context object.

The context manager keeps a reference to the first matching warning as the '`warning`' attribute; similarly, the '`filename`' and '`lineno`' attributes give you information about the line of Python

code from which the warning was triggered. This allows you to inspect the warning after the assertion:

```
with self.assertWarns(SomeWarning) as cm:  
    do_something()  
the_warning = cm.warning  
self.assertEqual(the_warning.some_attribute, 147)
```

`assertWarnsRegex(expected_warning, expected_regex, *args, **kwargs)`

Asserts that the message in a triggered warning matches a regexp. Basic functioning is similar to `assertWarns()` with the addition that only warnings whose messages also match the regular expression are considered successful matches.

Args:

`expected_warning`: Warning class expected to be triggered.
`expected_regex`: Regex (re.Pattern object or string) expected

to be found in error message.

`args`: Function to be called and extra positional args. `kwargs`: Extra kwargs. `msg`: Optional message used in case of failure. Can only be used

when `assertWarnsRegex` is used as a context manager.

`assert_(*kwargs)`

`countTestCases()`

`debug()`

Run the test without collecting errors in a `TestResult`

`defaultTestResult()`

classmethod `doClassCleanups()`

Execute all class cleanup functions. Normally called for you after `tearDownClass`.

`doCleanups()`

Execute all cleanup functions. Normally called for you after `tearDown`.

`fail(msg=None)`

Fail immediately, with the given message.

`failIf(**kwargs)`

`failIfAlmostEqual(**kwargs)`

`failIfEqual(**kwargs)`

`failUnless(**kwargs)`

`failUnlessAlmostEqual(**kwargs)`

`failUnlessEqual(**kwargs)`

`failUnlessRaises(**kwargs)`

`failureException`

alias of `AssertionError`

`id()`

`longMessage = True`

`maxDiff = 640`

`run(result=None)`

`setUp()`

Hook method for setting up the test fixture before exercising it.

classmethod `setUpClass()`

Hook method for setting up class fixture before running tests in the class.

`shortDescription()`

Returns a one-line description of the test, or None if no description has been provided.

The default implementation of this method returns the first line of the specified test method's docstring.

`skipTest(reason)`

Skip this test.

`subTest(msg=<object object>, **params)`

Return a context manager that will return the enclosed block of code in a subtest identified by the optional message and keyword parameters. A failure in the subtest marks the test case as failed but resumes execution at the end of the enclosed block, allowing further test code to be executed.

`tearDown()`

Hook method for deconstructing the test fixture after testing it.

`classmethod tearDownClass()`

Hook method for deconstructing the class fixture after running all tests in the class.

`test_create_large_file(fileName='largeFile.txt')`

`test_send_page_data()`

`createContentsMetaData(metaDataDictionary, contentFile)`

Creates a metadata file for use in uploading to AXON Args:

`metaDataDictionary`: Dictionary containing the fields and entries to the metadata file

Returns: Path to metadata file

defaultHash(*contentFile=None, BUFFER_SIZE=4096*)

findExecutable(*executable='', path=None*)

Tries to find 'executable' in the directories listed in 'path'. A string listing directories separated by 'os.pathsep'; defaults to os.environ['PATH']. Returns the complete filename or None if not found.

getHashFromContentMetaData(*metaDataFile*)

getIdentity()

loadJSON(*inJSONFile=None*)

readFileBytes(*fileInput=None, BUFFER_SIZE=4096*)

Reads an entire file and returns file bytes.

receiveCmd(*axonID=None, outputDirectory='/home/jdtarang/rad/raad/dox'*)

Sends AXON request to download a record based on the ID of the record
Args:

axonID[string]: the AXON Identification string that uniquely identifies a specific AXON record

Returns:

saveIdentityJSON(*identityFile=None*)

saveJSON(*saveFile=None, objectJSON=None*)

secureHash(*fileInput=None, BUFFER_SIZE=4096, secureMode='BLAKE2c'*)

sendAPI(*tarball=None, metadata=None*)

@summary: uses pyaxon to upload a record to Axon
@param tarball: the collected zip data which will be uploaded to Axon
@param metadata: the metadata dictionary associated with the packet. If None, packet will be extracted and the existing metadata file will be used.

`sendCmd(fileType=None, failureFile=None, metaDataFile=None,
contentFile=None)`

`sendInfo(fileType='intel-driveinfo-zip-v1', failureFile=None,
metaDataFile=None, contentFile=None)`

API for send Command function that returns the AXON ID for a successful upload
Args:

`fileType(str)`: file type of content file for upload
`failureFile(json)`: json file containing data about a failure encountered in the accompanying telemetry binary
`metaDataFile(json)`: metadata containing information about the content file
`contentFile(fileType)`: file containing a binary or binaries of telemetry data gained from some drive

Returns:

`success(bool)`: Contains whether the upload was successful
`line(str)`: Contains output line from Axon about AXON ID

`validateDownload(outputLocation, axonID)`

Function to return whether the received file is valid
Args:

`outputLocation: axonID:`

Returns:

`src.software.axon.axonInterface.hollowShell(options=None)`

`src.software.axon.axonInterface.main()`

`src.software.axon.axonInterface.setup(debug=True)`

```
class src.software.axon.packageInterface.Cipher_AES(key=None,  
iv=b'xff$&Gw\xb6fB\xd7y\n39\x97\xf7H', cipher_method=None,  
code_method=None, paddingMethod=None, debug=False)
```

Bases: `object`

`Cipher_MODE_CBC()`

`Cipher_MODE_ECB()`

```
decrypt(cipher_text, cipher_method=None, pad_method=None,  
code_method=None)
```

```
encrypt(text, cipher_method=None, pad_method=None,  
code_method=None)
```

`get_encryptedText()`

`get_iv()`

`get_key()`

`pad_default(y)`

`pad_pkcs5(y)`

`pad_user_defined(y, z)`

`set_iv(iv)`

`set_key(key)`

```
static testCase(msg, token, debug=False)
```

`testHarness()`

`text_verify(text, method=')`

`unpad_default()`

```
unpad_pkcs5()  
unpad_user_defined(z)  
  
src.software.axon.packageInterface.ExampleUsage(options)  
  
src.software.axon.packageInterface.compressTar(options, tarPack,  
tarFileName)  
  
src.software.axon.packageInterface.createTar(options)  
  
src.software.axon.packageInterface.hollowShell(options=None)  
  
src.software.axon.packageInterface.main()  
  
class src.software.axon.packageInterface.packageInterface(uid=-1,  
major=-1, minor=1, time=datetime.datetime(2022, 6, 29, 18, 37, 51, 330080), user='DarthVader', debug=True, absPath=None,  
binaryFileList=None, jsonFileList=None, timeSeriesFile=None,  
logFileList=None, pdfFileList=None, outFileName=None)
```

Bases: **object**

```
compressTarball(tarFile='compress.tar', mode='w:bz2',  
encodingType='utf-8')
```

```
createTarball(tarFileName='tarBall.tar', mode='w')
```

Mode Description r Opens a TAR file for reading. r: Opens a TAR file for reading with no compression. w or w: Opens a TAR file for writing with no compression. a or a: Opens a TAR file for appending with no compression. r:gz Opens a TAR file for reading with gzip compression. w:gz Opens a TAR file for writing with gzip compression. r:bz2 Opens a TAR file for reading with bzip2 compression. w:bz2 Opens a TAR file for writing with bzip2 compression.

```
createZIP(zipFileName='zipFile.zip')
```

Args:

zipFileName: Name of file that will hold the zipped directory

Returns:

None

decryptTarball(*dataToEncrypt='telemetry.bin.gpg'*,
outfile='telemetry.bin')

encryptTarball(*dataToEncrypt='telemetry.bin'*,
outfile='telemetry.bin.gpg')

extractTarball(*pathToFiles='extraction'*, *tarFile='compress.tar'*,
encodingType='utf-8')

userProfileCreate(*saveFile=None*, *email='joseph.d.tarango@intel.com'*,
passphrase='abc123', *gnupghome='/home/jdtarang/gpghome'*,
keyFile='mykeyfile.asc')

userProfileLoad(*profileFile=None*)

src.software.axon.packageInterface.setup(*debug=True*)

transformCSV.py

This module contains the basic functions for creating the content of a configuration file from CSV.

Args:

--inFile: Path for the configuration file where the time series data values CSV
--outFile: Path for the configuration file where the time series data values INI
--debug: Boolean flag to activate verbose printing for debug use

Example:

Default usage:

\$ python transformCSV.py

Specific usage:

```
$ python transformCSV.py  
--inFile C:
```

```
aadsrcsoftware ime-series.csv  
--outFile C:
```

```
aadsrcsoftware ime-series.ini  
--debug True
```

class
src.software.autoAI.transformCSV.TransformMetaData(*inputFileName=None*, *debug=False*, *transform=False*, *sectionName=None*, *outFolder=None*, *outFile='time-series-madness.ini'*)

Bases: **object**

CSVtoFrame(*inputFileName=None*)

analysisFrame = None

analysisFrameFormat = None

columnsList = None

debug = False

fileLocation = None

fileName = None

frameToINI(*analysisFrame=None*, *sectionName='Unknown'*, *outFolder=None*, *outFile='nil.ini'*)

getAnalysisFrame()

getAnalysisFrameFormat()

getColumnList()

static getDateParser(formatString='%Y-%m-%d %H:%M:%S.%f')

```
getHeaderFromFile(headerFilePath=None, method=1)  
getuniqueLists()  
uniqueLists = None  
  
src.software.autoAI.transformCSV.main()  
main function to be called when the script is directly executed from the  
command line
```

nlogPrediction.py

This module contains the basic functions for generating training data, instantiating all the stages, and training the nlog engine used for nlog event forecasting and failure prediction. This script requires

python 3, and tensorflow version 2.3.0 or higher.

Args:

--nlogFolder: Path for the nlog Folder in which the nlog event files are contained
--nlogParserFolder: Path for the folder in which the NLogFormats.py script is contained
--numComponents: Integer for the number of dimensions to be used in the NLOG description embeddings
--maxNumParams: Integer for the maximum number of parameters that can be contained in an NLOG description

for the specified formats file

--inputSize: Integer for the number of NLOG events to be considered as the input for the predictive models
--maxOutputSize: Integer for the maximum number of NLOG events to be predicted with the models
--debug: Verbose printing for debug use
--widthModelType: name of the model type to be used in the linear regression model for determining the number

of NLOG events to be predicted. Must be selected from the following: ['elastic', 'lasso', 'ridge', 'default']

--timeHiddenUnits: Integer for the number of neurons contained in each hidden layer for the NLOG time stamp
predictor model

--eventsHiddenUnits: Integer for the number of neurons contained in each hidden layer for the NLOG event
predictor model

--paramsHiddenUnits: Integer for the number of neurons contained in each hidden layer for the NLOG parameter
predictor mode

--eventsMaxEpochs: Integer for the maximum number of epochs to be considered when training the NLOG time stamp
predictor model

--eventsMaxEpochs: Integer for the maximum number of epochs to be considered when training the NLOG event
predictor model

--paramsMaxEpochs: Integer for the maximum number of epochs to be considered when training the NLOG parameter
predictor model

--timeOptimizer: name of the optimizer to be used in the NLOG time stamp predictor model. Must be selected

from the following: ['SGD', 'RMSprop', 'Adagrad', 'Adadelta',
'Adam', 'Adamax']

--eventsOptimizer: name of the optimizer to be used in the NLOG event predictor model. Must be selected

from the following: ['SGD', 'RMSprop', 'Adagrad', 'Adadelta',
'Adam', 'Adamax']

--paramsOptimizer: name of the optimizer to be used in the NLOG parameter predictor model. Must be selected

from the following: ['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax']

--timeLstmActivation: name of the activation function to be used in the LSTM layers of the NLOG time stamp predictor

model. Must be selected from the following:

['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']

--eventsLstmActivation: name of the activation function to be used in the LSTM layers of the NLOG event predictor

model. Must be selected from the following:

['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']

--paramsLstmActivation: name of the activation function to be used in the LSTM layers of the NLOG parameter

'predictor model. Must be selected from the following: ['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']

--timeLstmInitializer: name of the weight initializer function to be used in the LSTM layers of the NLOG time stamp

predictor model. Must be selected from the following:

['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']

--eventsLstmInitializer: name of the weight initializer function to be used in the LSTM layers of the NLOG event

predictor model. Must be selected from the following:

['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']

--paramsLstmInitializer: name of the weight initializer function to be used in the LSTM layers of the NLOG

parameter predictor model. Must be selected from the following:
['random_normal', 'random_uniform', 'truncated_normal', 'zeros',
'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal',
'constant', 'variance_scaling']

--timeDropout: Boolean flag that indicates if dropout in between layers should be applied to the NLOG time stamp predictor model

--eventsDropout: Boolean flag that indicates if dropout in between layers should be applied to the NLOG event predictor model

--paramsDropout: Boolean flag that indicates if dropout in between layers should be applied to the NLOG parameter predictor model

Example:

Default usage:

```
$ python nlogPrediction.py
```

Specific usage:

```
$ python nlogPrediction.py --nlogFolder data/output/nlog --  
nlogParserFolder software/parse/nlogParser  
--numComponents 50 --maxNumParams 8 --inputSize 4000 --  
maxOutputSize 1000 --debug True --widthModelType elastic --  
eventsHiddenUnits 128 --paramsHiddenUnits 128 --  
eventsMaxEpochs 2000 --paramsMaxEpochs 2000 --  
eventsOptimizer SGD --paramsOptimizer SGD --  
eventsLstmActivation relu --paramLstmActivation relu --  
eventsLstmInitializer random_normal --paramsLstmInitializer  
random_normal --eventsDropout True --paramsDropout True
```

class

```
src.software.autoAI.nlogPrediction.NlogDataProcessor(nlogFolder='data/o
```

utput/nlog', nlogParserFolder='software/parse/nlogParser', numComponents=50, maxNumParams=8, debug=False)

Bases: **object**

extractEventSignatures(*nlogEvents*, *maxListLength*=4000)

Function for extracting NLOG events encodings and parameter values from a list of NLOG event descriptions

Args:

nlogEvents: list of strings for the NLOG event descriptions
maxListLength: integer for the maximum length of the list to be considered as input for the neural network

Returns:

nlogEncodings: padded (or truncated) list of events encoding vectors
paramsList: padded (or truncated) list of parameter vectors
timeList: padded (or truncated) list of time vectors

extractFilesContent(*firstFile*, *secondFile*, *thirdFile*, *firstFileContent*=None, *secondFileContent*=None, *thirdFileContent*=None)

Function for extracting the content of three adjacent files if no content has been read or shifting the content for extracting the sliding window data

Args:

firstFile: string for the path of the first NLOG file containing the NLOG events to be read
secondFile: string for the path of the second NLOG file containing the NLOG events to be read
thirdFile: string for the path of the third NLOG file containing the NLOG events to be read
firstFileContent: None or dictionary mapping line content to line index in the first file
secondFileContent: None or dictionary mapping line content to line index in the second file
thirdFileContent: None or dictionary mapping line content to line index in the third file

Returns:

firstFileContent: dictionary mapping line content to line index in the first file or shifted content
secondFileContent: dictionary mapping line content to line index in the second file or shifted content
thirdFileContent: dictionary mapping line content to line index in the third file or shifted content

extractFilesFromFolder()

function for generating a list of nlog files to be used as inputs for data generation

Returns:

fileList: list of the names of the nlog files

extractInputDataForSingleFile(*fileOfInterest*, *timeDelta*, *inputSize*=4000)

Function for extracting input data from a specified NLOG file

Args:

fileOfInterest: string for the path of the NLOG file containing the NLOG events to be read
timeDelta: time delta to be used for prediction
inputSize: integer for the total number of NLOG events to be considered as inputs for the predictive models

Returns:

nlogDeltaTimes: list of time deltas (in seconds) between specified NLOG file and desired prediction range
nlogEventsInputs: list of NLOG event encodings to be used as input to the predictive model
nlogParamsInputs: list of NLOG parameter vectors to be used as input to the predictive model
nlogTimesInputs: list of NLOG time vectors to be used as input to the predictive model

generateDataSets(*inputSize*=4000, *maxOutputSize*=1000)

Function for generating the data sets to be fed into the neural network. Please bear in mind that the returned lists might still need to be converted to np.arrays and reshaped before being fed into any neural network model

Args:

inputSize: integer for the total number of NLOG events to be considered as inputs for the predictive models
maxOutputSize: integer for the total number of NLOG events to be considered in the prediction

Returns:

nlogDeltaTimes: list of time deltas (in seconds) between different NLOG files
nlogDeltaEventNums: list of number of events changed between different NLOG files
nlogEventsInputs: list of NLOG event encodings to be used as input to the predictive model
nlogEventsOutputs: list of NLOG event encodings to be used as reference output for the predictive model
nlogParamsInputs: list of NLOG parameter vectors to be used as input to the predictive model
nlogParamsOutputs: list of NLOG parameter vectors to be used as reference output to the predictive model
nlogTimesInputs: list of NLOG time stamp vectors to be used as input to the predictive model
nlogTimesOutputs: list of NLOG time stamp vectors to be used as reference output to the predictive model

generateDemoData(*inputSize=4000*)

Function for generating demo data to demonstrate the predictive capabilities of the system

Args:

inputSize: integer for the total number of NLOG events to be considered as inputs for the predictive models

Returns:

nlogDeltaTimes: list of time deltas (in seconds) between the last two NLOG files
nlogEventsInputs: list of NLOG event encodings to be used as input to the predictive model
nlogParamsInputs: list of NLOG parameter vectors to be used as input to the predictive model
nlogTimesInputs: list of NLOG time vectors to be used as input to the predictive model

firstOutput: list of NLOG descriptions unique to the last file
(expected output of predictive model)

*static paddEncodingList(encodingList,
maxDimensionForEncoding=50, maxListLength=4000)*

Function for padding (or truncating) the events encoding vector list so that it can be fed into a fixed input neural network

Args:

encodingList: list of variable length containing the events encoding vectors extracted so far
maxDimensionForEncoding: integer for the maximum number of dimensions used for the events encoding vectors
maxListLength: integer for the maximum length of the list to be considered as input for the neural network

Returns:

encodingList: padded (or truncated) list of events encoding vectors

*static paddParamList(paramList, maxNumParams=8,
maxListLength=4000)*

Function for padding (or truncating) the parameter vector list so that it can be fed into a fixed input neural network

Args:

paramList: list of variable length containing the parameter vectors extracted so far
maxNumParams: integer for the maximum number of parameters to be used for the parameter vector
maxListLength: integer for the maximum length of the list to be considered as input for the neural network

Returns:

paramsList: padded (or truncated) list of parameter vectors

static paddParams(params, maxNumParams=8)

Function for padding a parameter vector

Args:

 params: list of parameter values maxNumParams: integer for the maximum number of parameters to be used for the parameter vector

Returns:

 params: vector of parameter values with maxNumParams dimensionality

*static paddTimeList(*timeList*, *maxListLength*=4000)*

Function for padding (or truncating) the time vector list so that it can be fed into a fixed input neural network

Args:

timeList: list of variable length containing the time vectors extracted so far
 maxListLength: integer for the maximum length of the list to be considered as input for the neural network

Returns:

timeList: padded (or truncated) list of time vectors

*static readFileContentIntoDict(*tempFile*)*

Function for reading NLOG file content into a dictionary mapping line content to line index in the file

Args:

tempFile: string for the path of the NLOG file containing the NLOG events to be read

Returns:

tempFileContent: dictionary mapping line content to line index in the file for the specified file

class

src.software.autoAI.nlogPrediction.NlogEventsPredictor(*inputEventSeries*,
outputEventSeries)

Bases: **object**

```
class RNNUtility
```

Bases: `object`

utility class containing functions for processing a configuration file and loading it into a dictionary

```
static compileAndFit(model, inputData, outputData, patience=20,  
maxEpochs=2000, optimizer='Adam')
```

function for configuring a model, compiling it, and fitting it to the specified training and validation data

Args:

model: Tensorflow model to be compiled and fitted

inputData: list of 2D np.arrays containing the encodings for the event descriptions to be used as inputs

to the model

outputData: list of 2D np.arrays containing the encodings for the event descriptions to be used as

reference outputs for training

patience: Number of epochs with no improvement after which training will be stopped. maxEpochs: Number of trials to be run optimizer: optimizer to be used for the model, must be selected from the following

```
['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',  
'Adamax']
```

Returns:

history: list of loss values and metrics for each epoch model: Tensorflow model with the new trained weights

```
genericPredictor(hiddenUnits, maxEpochs=2000, optimizer='Adam',  
lstm_activation='tanh', lstm_initializer='glorot_uniform',  
dropout=False)
```

wrapper function for generating, training, and evaluating an Nlog event predictive model

Args:

hiddenUnits: Integer for the number of neurons contained in each hidden layer
maxEpochs: Integer for the maximum number of epochs to be considered when training the model
optimizer: optimizer to be used for the model, must be selected from the following

`['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',
'Adamax']`

`lstm_activation`: activation function for LSTM layers. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',
'elu', 'exponential']`

`lstm_initializer`: weight initializer for the lstm layers. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal',
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',
'orthogonal', 'constant', 'variance_scaling']`

`dropout`: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

`model`: Tensorflow NLOG event predictive model

`predictionEmbeddings`: list of 2D np.arrays containing the encodings for the event descriptions predicted by

the model

`sequentialNet(hiddenUnits, lstm_activation='tanh',
lstm_initializer='glorot_uniform', dropout=False)`

function for creating a generic RNN model using the sequential keras architecture

Args:

hiddenUnits: Integer for the number of neurons contained in each hidden layer
lstm_activation: activation function for LSTM layers. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']`

lstm_initializer: weight initializer for the lstm layers. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']`

dropout: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

`lstmModel`: Tensorflow RNN model containing the desired topology

class

`src.software.autoAI.nlogPrediction.NlogParameterPredictor(inputParamSeries, outputParamSeries)`

Bases: `object`

class `RNNUtility`

Bases: `object`

utility class containing functions for processing a configuration file and loading it into a dictionary

static `compileAndFit(model, inputData, outputData, patience=20, maxEpochs=2000, optimizer='Adam')`

function for configuring a model, compiling it, and fitting it to the specified training and validation data

Args:

model: Tensorflow model to be compiled and fitted

inputData: list of 2D np.arrays containing the parameter vectors from the event descriptions to be used

as inputs to the model

outputData: list of 2D np.arrays containing the parameter vectors for the event descriptions to be used

as reference outputs for training

patience: Number of epochs with no improvement after which training will be stopped. maxEpochs: Number of trials to be run optimizer: optimizer to be used for the model, must be selected from the following

`['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',
'Adamax']`

Returns:

history: list of loss values and metrics for each epoch model:
Tensorflow model with the new trained weights

`genericPredictor(hiddenUnits, maxEpochs=2000, optimizer='Adam',
lstm_activation='tanh', lstm_initializer='glorot_uniform',
dropout=False)`

wrapper function for generating, training, and evaluating an Nlog parameter predictive model

Args:

hiddenUnits: Integer for the number of neurons contained in each hidden layer

maxEpochs: Integer for the maximum number of epochs to be considered when training the model

optimizer:

optimizer to be used for the model, must be selected from the following

```
['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',  
 'Adamax']
```

lstm_activation: activation function for LSTM layers. Must be selected from the following:

```
['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',  
 'elu', 'exponential']
```

lstm_initializer: weight initializer for the lstm layers. Must be selected from the following:

```
['random_normal', 'random_uniform', 'truncated_normal',  
 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',  
 'orthogonal', 'constant', 'variance_scaling']
```

dropout: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

model: Tensorflow NLOG parameter predictive model

predictionEmbeddings: list of 2D np.arrays containing the parameter vectors for the event descriptions

predicted by the model

`sequentialNet(hiddenUnits, lstm_activation='tanh',
lstm_initializer='glorot_uniform', dropout=False)`

function for creating a generic RNN model using the sequential keras architecture

Args:

hiddenUnits: Integer for the number of neurons contained in each hidden layer
lstm_activation: activation function for LSTM layers. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']`

`lstm_initializer`: weight initializer for the lstm layers. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']`

`dropout`: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

`LstmModel`: Tensorflow RNN model containing the desired topology

class

`src.software.autoAI.nlogPrediction.NlogPredictor(nlogFolder='data/output/nlog', nlogParserFolder='software/parse/nlogParser',
numComponents=50, maxNumParams=8, inputSize=4000,
maxOutputSize=1000, debug=False)`

Bases: `object`

`eventPredictorAPI(eventsHiddenUnits=128, eventsMaxEpochs=2000,
eventsOptimizer='Adam', eventsLstmActivation='tanh',
eventsLstmInitializer='glorot_uniform', eventsDropout=False)`

Function for generating, training and evaluating a Nlog event predictive model

Args:

`eventsHiddenUnits`: Integer for the number of neurons contained in each hidden layer
`eventsMaxEpochs`: Integer for the maximum number of epochs to be considered when training the model
`eventsOptimizer`: optimizer to be used for the model, must be selected from the following

`['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax']`

`eventsLstmActivation`: activation function for LSTM layers.
Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']`

`eventsLstmInitializer`: weight initializer for the lstm layers. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']`

`eventsDropout`: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

`eventPredictorModel`: Tensorflow NLOG event predictive model
`eventPredictorAccuracy`: accuracy score for the number of predicted events that were actually correctly

predicted

`generateOptimizer(optimizer='Adam', learningRate=0.1)`

function for generating an optimizer with a specified learning rate

Args:

`optimizer`: optimizer to be used for the model, must be selected from the following

`['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax']`

`learningRate`: float value for the learning rate to be used in the model

Returns:

`opt`: optimizer structure with specified learning rate

```
nlogPredictorAPI(widthModelType='elastic', timeHiddenUnits=128,  
timeMaxEpochs=2000, timeOptimizer='Adam',  
timeLstmActivation='tanh', timeLstmInitializer='glorot_uniform',  
timeDropout=False, eventsHiddenUnits=128, eventsMaxEpochs=2000,  
eventsOptimizer='Adam', eventsLstmActivation='tanh',  
eventsLstmInitializer='glorot_uniform', eventsDropout=False,  
paramsHiddenUnits=128, paramsMaxEpochs=2000,  
paramsOptimizer='Adam', paramsLstmActivation='tanh',  
paramsLstmInitializer='glorot_uniform', paramsDropout=False)
```

Function for executing the entrie NLOG prediction system

Args:

timeDropout: timeLstmActivation: timeLstmInitializer:

timeMaxEpochs: timeHiddenUnits: timeOptimizer:

widthModelType: name of the model type to be used in the
linear regression model for determining the number

of NLOG events to be predicted. Must be selected from the
following: ['elastic', 'lasso', 'ridge', 'default']

eventsHiddenUnits: Integer for the number of neurons contained
in each hidden layer for the event predictive
model

eventsMaxEpochs: Integer for the maximum number of epochs
to be considered when training the event
predictive model

eventsOptimizer: optimizer to be used for the event predictive
model, must be selected from the following:

['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax']

eventsLstmActivation: activation function for LSTM layers in
the event predictive model. Must be selected

from the following: ['relu', 'sigmoid', 'softmax', 'softplus',
'softsign', 'tanh', 'selu', 'elu', 'exponential']

eventsLstmInitializer: weight initializer for the lstm layers in the event predictive model. Must be

selected from the following: ['random_normal',
'random_uniform', 'truncated_normal', 'zeros', 'ones',
'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal',
'constant', 'variance_scaling']

eventsDropout: Boolean flag to indicate if dropout should be applied in between layers in the event predictive model

paramsHiddenUnits: Integer for the number of neurons contained in each hidden layer for the parameter predictive model

paramsMaxEpochs: Integer for the maximum number of epochs to be considered when training the parameter predictive model

paramsOptimizer: optimizer to be used for the parameter predictive model, must be selected from the following: ['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax']

paramsLstmActivation: activation function for LSTM layers in the parameter predictive model. Must be

selected from the following: ['relu', 'sigmoid', 'softmax',
'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']

paramsLstmInitializer: weight initializer for the lstm layers in the parameter predictive model. Must be

selected from the following: ['random_normal',
'random_uniform', 'truncated_normal', 'zeros', 'ones',
'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal',
'constant', 'variance_scaling']

paramsDropout: Boolean flag to indicate if dropout should be applied in between layers in the parameter

predictive model

Returns:

widthPredictorModel: linear regression model to be used for prediction
eventPredictorModel: Tensorflow NLOG event predictive model
paramsPredictorModel: Tensorflow NLOG parameter predictive model

*paramPredictorAPI(paramsHiddenUnits=128,
paramsMaxEpochs=2000, paramsOptimizer='Adam',
paramsLstmActivation='tanh', paramsLstmInitializer='glorot_uniform',
paramsDropout=False)*

Function for generating, training and evaluating a Nlog parameter predictive model

Args:

paramsHiddenUnits: Integer for the number of neurons contained in each hidden layer
paramsMaxEpochs: Integer for the maximum number of epochs to be considered when training the model
paramsOptimizer: optimizer to be used for the model, must be selected from the following

['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',
'Adamax']

paramsLstmActivation: activation function for LSTM layers.
Must be selected from the following:

['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',
'elu', 'exponential']

paramsLstmInitializer: weight initializer for the lstm layers.
Must be selected from the following:

['random_normal', 'random_uniform', 'truncated_normal',
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',
'orthogonal', 'constant', 'variance_scaling']

paramsDropout: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

paramsPredictorModel: Tensorflow NLOG parameter predictive model
paramPredictorAccuracy: accuracy score for the number of predicted parameters that were actually correctly

predicted

timePredictorAPI(*timeHiddenUnits*=128, *timeMaxEpochs*=2000,
timeOptimizer='Adam', *timeLstmActivation*='tanh',
timeLstmInitializer='glorot_uniform', *timeDropout*=False)

Function for generating, training and evaluating a Nlog event predictive model

Args:

timeHiddenUnits: Integer for the number of neurons contained in each hidden layer
timeMaxEpochs: Integer for the maximum number of epochs to be considered when training the model
timeOptimizer: optimizer to be used for the model, must be selected from the following

['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',
'Adamax']

timeLstmActivation: activation function for LSTM layers. Must be selected from the following:

['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',
'elu', 'exponential']

timeLstmInitializer: weight initializer for the lstm layers. Must be selected from the following:

['random_normal', 'random_uniform', 'truncated_normal',
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',
'orthogonal', 'constant', 'variance_scaling']

timeDropout: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

timePredictorModel: Tensorflow NLOG event predictive model

timePredictorAccuracy: accuracy score for the number of predicted events that were actually correctly

predicted

widthPredictorAPI(widthModelType='elastic')

Function for generating, training and evaluating a width predictive model

Args:

widthModelType: name of the model type to be used in the linear regression model for determining the number of NLOG events to be predicted. Must be selected from the following: ['elastic', 'lasso', 'ridge', 'default']

Returns:

widthPredictorModel: linear regression model to be used for prediction
widthPredictorScore: R^2 score for the correlation between time deltas and number of events changed

widthPredictorAccuracy: accuracy score for the number of predicted changes that was within 5% of the right

value

class src.software.autoAI.nlogPrediction.NlogTimePredictor(inputSeries, outputSeries)

Bases: **object**

class RNNUtility

Bases: **object**

utility class containing functions for processing a configuration file and loading it into a dictionary

static compileAndFit(model, inputData, outputData, patience=20, maxEpochs=2000, optimizer='Adam')

function for configuring a model, compiling it, and fitting it to the specified training and validation data

Args:

model: Tensorflow model to be compiled and fitted

inputData: np.array with the time stamps to be used for the input of the neural network outputData: np.array with the time stamps to be used as reference output for the neural

network patience: Number of epochs with no improvement after which training will be stopped. maxEpochs: Number of trials to be run optimizer: optimizer to be used for the model, must be selected from the following

`['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',
'Adamax']`

Returns:

history: list of loss values and metrics for each epoch model: Tensorflow model with the new trained weights

*genericPredictor(hiddenUnits, maxEpochs=2000, optimizer='Adam',
lstm_activation='tanh', lstm_initializer='glorot_uniform',
dropout=False)*

wrapper function for generating, training, and evaluating an Nlog Time predictive model

Args:

hiddenUnits: Integer for the number of neurons contained in each hidden layer maxEpochs: Integer for the maximum number of epochs to be considered when training the model optimizer: optimizer to be used for the model, must be selected from the following

`['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',
'Adamax']`

`lstm_activation`: activation function for LSTM layers. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',
'elu', 'exponential']`

`lstm_initializer`: weight initializer for the lstm layers. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal',
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',
'orthogonal', 'constant', 'variance_scaling']`

`dropout`: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

`model`: Tensorflow NLOG time predictive model

`predictionEmbeddings`: list of 2D np.arrays containing the vectors for the time stamps predicted by

the model

`sequentialNet(hiddenUnits, lstm_activation='tanh',
lstm_initializer='glorot_uniform', dropout=False)`

function for creating a generic RNN model using the sequential keras architecture

Args:

`hiddenUnits`: Integer for the number of neurons contained in each hidden layer
`lstm_activation`: activation function for LSTM layers. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',
'elu', 'exponential']`

`lstm_initializer`: weight initializer for the lstm layers. Must be selected from the following:

```
['random_normal', 'random_uniform', 'truncated_normal',
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',
'orthogonal', 'constant', 'variance_scaling']
```

`dropout`: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

`lstmModel`: Tensorflow RNN model containing the desired topology

class

```
src.software.autoAI.nlogPrediction.NlogTokenizer(nlogParserFolder='software/parse/nlogParser', numComponents=50, debug=False)
```

Bases: `object`

`decodeEventEncoding(nlogSignature)`

Function for decoding an event encoding to obtain the formatting string and the number of parameters to be used

Args:

`nlogSignature`: dimensionality-reduced vector produced by the predictor model

Returns:

`nlogEncoding[0]`: encoding for closest neighbor to
`nlogSignature parseString`: formatting string associated with closest neighbor to `nlogSignature`
`paramNum`: number of parameters to be included in the formatting string

`encodeNlogEvents()`

Function for reading NLOG events and generating encodings for each of their descriptions based on the natural language used in them. Encodings are generated utilizing a pretrained BERT model

Returns:

labelsToDescriptions: dictionary mapping labels (first ten characters of description) to a tuple containing relevant details about the NLOG event parsing and pattern matching to it

encodingsToParsers: dictionary of description encodings to formatting strings for NLOG descriptions
encodings: list of encodings for NLOG event descriptions
labelIndices: list of tuples containing the label associated with each NLOG event and the index that such

event occupies inside the list of events inside that label collection

getSignaturesNeighbors()

Function for initializing a nearest neighbor model to match event signatures (encodings) with the list of events encodings generated from the list of events contained in NLogFormats.py

Returns:

neighbors: NearestNeighbors instance to be used for obtaining the closest neighbor to a given encoding

obtainEventSignature(*nlogDescription=None*)

Function for obtaining an event encoding based on the given description

Args:

nlogDescription: string for the NLOG event

Returns:

nlogEncoding: dimensionality-reduced vector representing the NLOG description
params: list of parameter values extracted from the specified *nlogDescription*
finalPos: index of the NLOG event as specified in NLogFormats.py to be used as input for the parameter

`predictor`

`reduceDimensionalityOfEmbeddings(numComponents=50)`

Function for reducing the vector dimensionality for the description encodings utilizing PCA

Args:

numComponents: number of dimensions for the output encodings

Returns:

`reformatString(formatStr, params)`

Function for handling some special formatting issues -- differences between Python and FW for printf.

Stolen from nlogpost2.py reformat

Args:

formatStr: Starting format string
 params: Parameter value tuple

Returns:

string: Updated format string
 tuple: New parameter tuple

`replaceFormattingForRegex(matchobj)`

Function for replacing a format specifier with its corresponding regex pattern

Args:

matchobj: formatting specifier extracted from the original formatting string for an NLOG event

Returns:

repStr: regex expression to capture potential values written by the format specifier

`class src.software.autoAI.nlogPrediction.NlogUtils`

Bases: `object`

`static add_value_labels(ax, spacing=5)`

Add labels to the end of each bar in a bar chart.

Arguments:

`ax` (`matplotlib.axes.Axes`): The `matplotlib` object containing the axes
 of the plot to annotate.

`spacing` (int): The distance between the labels and the bars.

`static extractNlogComponents(lineComponents, initialPos)`

function for extracting the NLOG components from the list of strings generated after splitting a line from the original NLOG files

Args:

`lineComponents`: list of strings containing all the NLOG fields after they have been separated using
 blank spaces

`initialPos`: integer for the position that contains the date element

Returns:

`nlogComponents`: list of NLOG fields with the right format for processing

`extractNlogTable(nlogFilePath)`

Function for generating the list of elements to be included in the report's NLOG table

Args:

`nlogFilePath`: Path to the NLOG file that contains the summarized NLOG events

Returns:

nlogTable: list of rows to be included in the NLOG table of the report

static extractNlogTime(line)

function for extracting the date from on NLOG event line

Args:

line: string for an NLOG event extracted from the parsed NLOG text files

Returns:

list of floats for [hours, minutes, seconds]

static formatNlogTime(time)

Function for generating a string representation for the time

Args:

time: list floats for [hours, minutes, seconds]

Returns:

timeStr: string with the string representation for the time

static getDateFromFileNames(fileName='sample')

function to extract datetime object from directory name. File naming must follow the right format for this function to work (e.g. tag_date.txt)

Args:

fileName: String representation of the file name

Returns:

datetime object containing the date parsed in the dirName

static splitDataSet(x, y, testSize=0.1, randomState=None)

Function for splitting the dataset into testing and training data sets

Args:

x: input values y: output values testSize: fraction of values to be used for the testing set randomState: integer denoting random seed for consistency across trials

Returns:

xTrain: numpy array of x values to be used for training xTest: numpy array of x values to be used for testing yTrain: numpy array of y values to be used for training yTest: numpy array of x values to be used for testing

class

src.software.autoAI.nlogPrediction.NlogWidthPredictor(*timeDeltaSeries*,
eventDeltaSeries)

Bases: **object**

defaultLinearModel()

function for creating a generic linear model

Returns:

clf: linear regression model to be used for prediction score: R^2 score for the correlation between time deltas and number of events changed

elasticNetLinearModel()

function for creating a generic ElasticNet linear model with cross validation

Returns:

reg: elastic net linear regression model to be used for prediction score: R^2 score for the correlation between time deltas and number of events changed

genericLinearPredictor(*modelType='elastic'*)

function for selecting and training a linear regression model for number of event changes. It also predicts output for the testing set

Args:

modelType: name of the model type to be used in the linear regression model for determining the number of NLOG events to be predicted. Must be selected from the following: ['elastic', 'lasso', 'ridge', 'default']

Returns:

model: linear regression model to be used for prediction score: R^2 score for the correlation between time deltas and number of events changed predictions: list of number of predicted number of event changes for each time delta in the testing set

lassoLinearModel()

function for creating a generic Lasso linear model with cross validation

Returns:

reg: lasso linear regression model to be used for prediction score: R^2 score for the correlation between time deltas and number of events changed

ridgeLinearModel()

function for creating a generic Ridge linear model with cross validation

Returns:

clf: ridge linear regression model to be used for prediction score: R^2 score for the correlation between time deltas and number of events changed

src.software.autoAI.nlogPrediction.main()

main function to be called when the script is directly executed from the command line

NNStateBasedProcessing.py This module contains the basic functions for state based batching of NNs.

```
class  
src.software.autoAI.NNStateBasedProcessing.BigDataProcessing(debug=F  
alse, dataFilesList=None, partitionSize=65536,  
machineLearningTechnique=None)
```

Bases: `object`

`FSM_Action(state)`

`FSM_Transitions(state)`

`fileSize(currentFile)`

`static stateDictionary(state)`

`updateState()`

```
src.software.autoAI.NNStateBasedProcessing.main()
```

main function to be called when the script is directly executed from the command line

`mediaPredictionRNN.py`

This module contains the basic functions for loading the content of a configuration file and utilizing a RNN model to generate predictions on the time series values passed in as arguments into the main API. This script requires python 3, and tensorflow version 2.3.0 or higher.

Args:

--configFilePath: Path for the configuration file where the time series data values for the media errors are contained --targetObject: String for the name for the target Object to be considered for the time series prediction --targetFields: Comma-separated strings for the names of the object's fields to be considered for the time series prediction --plotColumn: String for the name of the target field to be considered for plotting the input and predictions --matrixProfile: Boolean flag to apply matrix profile to time series before using the RNN model --subSeqLen: Integer for the length of the sliding window for matrix profile (only

relevant if matrix profile flag is set) --inputWidth: Integer for the length of the input sequence to be considered for the prediction --labelWidth: Integer for the length of the output sequence expected from the prediction --shift: Integer for the shift to be considered when sliding the input window --batchSize: Integer for the size of the batch to be considered when generating data sets to be fed into the RNN model. --maxEpochs: Integer for the maximum number of epochs to be considered when training the model --hiddenUnits: Integer for the number of neurons contained in each hidden layer -- embeddedEncoding: Boolean flag to apply embedded encoding to time series as the first layer in the RNN model --categoricalEncoding: Boolean flag to apply label encoding to the time series values (usually used for categorical values) --lstm_activation: activation function for LSTM layers. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']`

--dense_activation: activation function for dense layer. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']`

--lstm_initializer: weight initializer for the lstm layers. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']`

--dense_initializer: weight initializer for the dense layer. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']`

--debug: Boolean flag to activate verbose printing for debug use

Example:

Default usage:

```
$ python mediaPredictionRNN.py
```

Specific usage:

```
$ python mediaPredictionRNN.py --configFilePath C:
```

aadsrcsoftware ime-series.ini

```
--targetObject NandStats --targetFields 01-biterrors,02-biterrors --
plotColumn 01-biterrors --matrixProfile True --subSeqLen 20 --
inputWidth 80 --LabelWidth 20 --shift 2 --hiddenUnits 32 --debug True
```

class

```
src.software.autoAI.mediaPredictionRNN.WindowGenerator(inputWidth,
labelWidth, shift, batchSize, trainDataStream, valDataStream,
testDataStream, labelColumns=None, embeddedEncoding=False)
```

Bases: `object`

property example

Get and cache an example batch of *inputs*, *labels* for plotting.

makeDataset(data, batchSize=32)

function for generating a timeseries dataset for the specified data

Args:

 data: DataFrame containing the values of interest
 batchSize:
 Integer for the size of the batch to be considered in each data set

Returns:

 ds: dataset containing pairs of inputs and outputs for the
 specified timeseries

*plot(model=None, encoders=None, plotColumn='02-biterrors',
maxSubplots=3, normalize=False, pdf=None)*

function for plotting example windows for a specified column

Args:

model: Tensorflow model containing the desired topology for producing the predictions encoders: Dictionary of label encoders for the data plotColumn: String for the name of the DataFrame column to be graphed maxSubplots: Integer for the maximum number of subplots to be graphed normalize: Boolean flag to indicate whether the data has been normalized (and therefore the encoders should

not be used)

pdf: pdf file name to save.

Returns:

`plotEmbedded(numFeatures, encoders=None, model=None,
plotColumn='02-biterrors', maxSubplots=3, pdf=None)`

function for plotting a specified window for an embedded model

Args:

numFeatures: Integer for the number of object fields included in the model encoders: Dictionary of label encoders for the data model: tensorflow model containing the desired embedded and RNN topology for producing the predictions plotColumn: String for the name of the field to be plotted maxSubplots: Integer for the maximum number of subplots to be generated pdf: pdf file name to save.

Returns:

`splitWindow(features)`

function for splitting a window into two independent DataFrames, one for input and one for labels (expected results)

Args:

features: DataFrame containing all the data values for a set

Returns:

inputs: DataFrame with the data values for inputs
labels: DataFrame with the data values for labels (expected results)

property test

property train

property val

src.software.autoAI.mediaPredictionRNN.add_value_labels(*ax*, *spacing*=5)

Add labels to the end of each bar in a bar chart.

Arguments:

ax (matplotlib.axes.Axes): The matplotlib object containing the axes of the plot to annotate.

spacing (int): The distance between the labels and the bars.

src.software.autoAI.mediaPredictionRNN.main()

main function to be called when the script is directly executed from the command line

class

src.software.autoAI.mediaPredictionRNN.mediaPredictionRNN(*targetFields*)

Bases: **object**

class

src.software.autoAI.mediaPredictionRNN.timeSeriesRNN(*configFilePath*,
matrixProfileFlag=False, *subSeqLen=20*, *embeddedEncoding=False*,
categoricalEncoding=False, *debug=True*)

Bases: **object**

class RNNUtility

Bases: **object**

utility class containing functions for processing a configuration file and loading it into a dictionary

static compileAndFit(model, window, patience=5, maxEpochs=200, optimizer='Adam')

function for configuring a model, compiling it, and fitting it to the specified training and validation data

Args:

model: Tensorflow model to be compiled and fitted
window: window instance containing the training and validation data
patience: Number of epochs with no improvement after which training will be stopped.
maxEpochs: Number of trials to be run
optimizer: optimizer to be used for the model, must be selected from the following

`['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',
'Adamax']`

Returns:

history: Fitted Tensorflow model

static check_stationarity(timeseriesCandidate, debug: bool = False)

Augmented Dickey-Fuller (ADF) test can be used to test the null hypothesis that the series is non-stationary. The ADF test helps to understand whether a change in Y is a linear trend or not. If there is a linear trend but the lagged value cannot explain the change in Y over time, then our data will be deemed non-stationary. The value of test statistics is less than 5% critical value and p-value is also less than 0.05 so we can reject the null hypothesis and Alternate Hypothesis that time series is Stationary seems to be true. When there is nothing unusual about the time plot and there appears to be no need to do any data adjustments. There is no evidence of changing variance also so we will not do a Box-Cox transformation.

Args:

timeseriesCandidate: Time series
debug: developer debug flag

Returns: Determination of whether the data is stationary or not?

`dataIndependentEncoding(inputDataStream=None,
inputDataLabels=None)`

function for generating a numerical encoding for the input passed in as the `inputDataStream`

Args:

`inputDataStream`: DataFrame containing the values to be encoded
`inputDataLabels`: List of strings containing the name of the fields to be processed

Returns:

`encoders`: Dictionary with the training encoders for the specified DataFrame fields
`inputEncodings`: DataFrame with all the encoding results

`decomposeDataSets(trainDataStream, valDataStream, testDataStream,
targetFields, inputWidth, labelWidth, shift, batchSize)`

function for decomposing the entire train, validation, and testing data sets into independent lists for inputs and outputs

Args:

`trainDataStream`: DataFrame containing all the training data
`valDataStream`: DataFrame containing all the validation data
`testDataStream`: DataFrame containing all the testing data
`targetFields`: List of strings for the names of the object's fields to be considered for the time

series prediction

`inputWidth`: Integer for the length of the input sequence to be considered for the prediction
`labelWidth`: Integer for the length of the output sequence expected from the prediction
`shift`: Integer for the shift to be considered when sliding the input window
`batchSize`: Integer for the size of the batch to be considered when generating data sets to be fed into the RNN

model

Returns:

trainInputData: List containing all the training input data
trainOutputData: List containing all the training output data
valInputData: List containing all the validation input data
valOutputData: List containing all the validation output data
testInputData: List containing all the testing input data
testOutputData: List containing all the testing output data

`embeddedPredictor(multiPerformance, inputWidth, labelWidth, shift,
batchSize, hiddenUnits, numFeatures, DataStream, trainDataStream,
valDataStream, testDataStream, plotColumn, targetFields, encoders,
maxEpochs=2000, pdf=None, optimizer='Adam',
lstm_activation='tanh', dense_activation='sigmoid',
lstm_initializer='glorot_uniform', dense_initializer='glorot_uniform',
dropout=False)`

function for generating a window and an embedded RNN model for a given data set, and training the model on the specified data

Args:

multiPerformance: Dictionary where the performance metric results will be stored
inputWidth: Integer for the length of the input sequence to be considered for the prediction
labelWidth: Integer for the length of the output sequence expected from the prediction
shift: Integer for the shift to be considered when sliding the input window
batchSize: Integer for the size of the batch to be considered when generating data sets to be fed into the RNN

model

hiddenUnits: Integer for the number of neurons contained in each hidden layer
numFeatures: Integer for the number of object fields included in the model
DataStream: DataFrame containing the entirety of the population space
trainDataStream: DataFrame containing all the training data
valDataStream: DataFrame

containing all the validation data testDataStream: DataFrame containing all the testing data plotColumn: String for the name of the target field to be considered for plotting the input and predictions targetFields: List of strings for the names of the object's fields to be considered for the time

series prediction

encoders: Dictionary of label encoders for the data maxEpochs: Integer for the maximum number of epochs to be considered when training the model pdf: pdf file name to save. optimizer: optimizer to be used for the model, must be selected from the following

`['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',
'Adamax']`

lstm_activation: activation function for LSTM layers. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',
'elu', 'exponential']`

dense_activation: activation function for dense layer. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',
'elu', 'exponential']`

lstm_initializer: weight initializer for the lstm layers. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal',
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',
'orthogonal', 'constant', 'variance_scaling']`

dense_initializer: weight initializer for the dense layer. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal',
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',
'orthogonal', 'constant', 'variance_scaling']`

dropout: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

multi_lstm_model: Tensorflow embedded RNN model containing the desired topology

embeddingNet(*inputEncodings*, *targetFields*, *hiddenUnits*, *inputWidth*,
labelWidth, *numFeatures*, *batchSize*, *lstm_activation='tanh'*,
dense_activation='sigmoid', *lstm_initializer='glorot_uniform'*,
dense_initializer='glorot_uniform', *dropout=False*)

function for creating an embedded RNN model

Args:

inputEncodings: DataFrame containing numerical values to be turned into vectors through the embedded encoding

targetFields: List of strings for the names of the object's fields to be considered for the time series prediction

hiddenUnits: Integer for the number of neurons contained in each hidden layer
inputWidth: Integer for the length of the input sequence to be considered for the prediction
labelWidth: Integer for the length of the output sequence expected from the prediction
numFeatures: Integer for the number of object fields included in the model
batchSize: Integer for the size of the batch to be considered when generating data sets to be fed into'

the RNN model

lstm_activation: activation function for LSTM layers. Must be selected from the following:

['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']

dense_activation: activation function for dense layer. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']`

lstm_initializer: weight initializer for the lstm layers. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']`

dense_initializer: weight initializer for the dense layer. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']`

dropout: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

model: Tensorflow model containing the indicated embedded layers and LSTMs

formatDataSets(*inputData*, *outputData*, *numFeatures*)

function for adjusting the shape of input and output lists so that they can be fed into the RNN model

Args:

inputData: List containing input data
outputData: List containing output data
numFeatures: Integer for the number of object fields included in the model

Returns:

inputList: List of input data values in the right shape to be fed into the RNN model
expectedResults: List of output data values in the right shape to be fed into the RNN model

`generateDecompElemLists(initialList, inputData, outputData,
inputWidth, labelWidth, batchSize)`

function for decomposing the initial tuple list into two independent lists of inputs and outputs with the desired shapes

Args:

initialList: List of tuples (input, output) containing the desired data
inputData: List where the resulting input sequences will be stored
outputData: List where the resulting output sequences will be stored
inputWidth: Integer for the length of the input sequence to be considered for the prediction
labelWidth: Integer for the length of the output sequence expected from the prediction
batchSize: Integer for the size of the batch to be considered when generating data sets to be fed into the RNN

model

Returns:

`generateNormalizedWindow(trainDataStream, valDataStream,
testDataStream, inputWidth, labelWidth, shift, batchSize)`

function for generating a window containing normalized data for a generic model

Args:

trainDataStream: DataFrame containing all the training data
valDataStream: DataFrame containing all the validation data
testDataStream: DataFrame containing all the testing data
inputWidth: Integer for the length of the input sequence to be considered for the prediction
labelWidth: Integer for the length of the output sequence expected from the prediction
shift: Integer for the shift to be considered when sliding the input window
batchSize: Integer for the size of the batch to be considered when generating data sets to be fed into the RNN

model

Returns:

wideWindow: WindowGenerator instance containing normalized data

genericPredictor(*multiPerformance*, *inputWidth*, *labelWidth*, *shift*,
batchSize, *hiddenUnits*, *numFeatures*, *trainDataStream*, *valDataStream*,
testDataStream, *plotColumn*, *encoders*, *maxEpochs*=200, *pdf*=None,
optimizer='Adam', *lstm_activation*='tanh', *dense_activation*='sigmoid',
lstm_initializer='glorot_uniform', *dense_initializer*='glorot_uniform',
dropout=False)

function for generating a window and a generic RNN model for a given data set, and training the model on the specified data

Args:

multiPerformance: Dictionary where the performance metric results will be stored
inputWidth: Integer for the length of the input sequence to be considered for the prediction
labelWidth: Integer for the length of the output sequence expected from the prediction
shift: Integer for the shift to be considered when sliding the input window
batchSize: Integer for the size of the batch to be considered when generating data sets to be fed into the RNN

model

hiddenUnits: Integer for the number of neurons contained in each hidden layer
numFeatures: Integer for the number of object fields included in the model
trainDataStream: DataFrame containing all the training data
valDataStream: DataFrame containing all the validation data
testDataStream: DataFrame containing all the testing data
plotColumn: String for the name of the target field to be considered for plotting the input and predictions
encoders: Dictionary of label encoders for the data
maxEpochs: Integer for the maximum number of epochs to be considered when training the model
pdf: Save data graph to PDF.
optimizer: optimizer to be used for the model, must be selected from the following

`['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',
'Adamax']`

`lstm_activation`: activation function for LSTM layers. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',
'elu', 'exponential']`

`dense_activation`: activation function for dense layer. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',
'elu', 'exponential']`

`lstm_initializer`: weight initializer for the lstm layers. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal',
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',
'orthogonal', 'constant', 'variance_scaling']`

`dense_initializer`: weight initializer for the dense layer. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal',
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',
'orthogonal', 'constant', 'variance_scaling']`

`dropout`: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

`multi_lstm_model`: Tensorflow generic RNN model containing the desired topology

`loadDictIntoDataFrames(targetObject='NandStats', targetFields=None)`
function for loading a configuration file into a pandas DataFrame to be used as input for the neural network

Args:

targetObject: String for the name for the target Object to be considered for the time series prediction
targetFields: List of strings for the names of the object's fields to be considered for the time series prediction

Returns:

dataStream: DataFrame containing the entirety of the population space
trainDataStream: DataFrame containing all the training data
valDataStream: DataFrame containing all the validation data
testDataStream: DataFrame containing all the testing data
encoders: Dictionary for label encoders for each object field

partitionDataFrame(*targetFields=None*)

function for dividing the dataframe into train, validation, and testing sets

Args:

targetFields: Fields inside the Data Frame to be considered for the RNN model

Returns:

dataStream: DataFrame containing the entirety of the population space
trainDataStream: DataFrame containing all the training data
valDataStream: DataFrame containing all the validation data
testDataStream: DataFrame containing all the testing data
encoders: Dictionary for label encoders for each object field

plotPerformanceMetrics(*multiPerformance, multi_lstm_model, pdf=None*)

function for plotting the performance metric results contained in multiPerformance as a bar graph

Args:

multiPerformance: Dictionary containing the model name and the metric result
multi_lstm_model: Tensorflow model from

which the performance metrics were obtained pdf: Save data graph to PDF.

Returns:

```
sequentialNet(hiddenUnits, labelWidth, numFeatures,  
lstm_activation='tanh', dense_activation='sigmoid',  
lstm_initializer='glorot_uniform', dense_initializer='glorot_uniform',  
dropout=False)
```

function for creating a generic RNN model

Args:

hiddenUnits: Integer for the number of neurons contained in each hidden layer
labelWidth: Integer for the length of the output sequence expected from the prediction
numFeatures: Integer for the number of object fields included in the model
lstm_activation: activation function for LSTM layers. Must be selected from the following:

```
['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',  
'elu', 'exponential']
```

dense_activation: activation function for dense layer. Must be selected from the following:

```
['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',  
'elu', 'exponential']
```

lstm_initializer: weight initializer for the lstm layers. Must be selected from the following:

```
['random_normal', 'random_uniform', 'truncated_normal',  
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',  
'orthogonal', 'constant', 'variance_scaling']
```

dense_initializer: weight initializer for the dense layer. Must be selected from the following:

```
['random_normal', 'random_uniform', 'truncated_normal',  
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',
```

'orthogonal', 'constant', 'variance_scaling']

dropout: Boolean flag to indicate if dropout should be applied in between layers in the model

Returns:

multi_lstm_model: Tensorflow RNN model containing the desired topology

setCategoricalEncoding(*value=True*)

function for setting the categoricalEncoding flag

Args:

value: Boolean value for the categoricalEncoding flag

Returns:

setEmbeddedEncoding(*value=True*)

function for setting the embeddedEncoding flag

Args:

value: Boolean value for the embeddedEncoding flag

Returns:

setMatrixProfile(*value, subSeqLen=20*)

function for setting the matrixProfileFlag and generating the matrix profile dataDict if the subSeqLen has changed or none has been generated before

Args:

value: Boolean value for matrixProfileFlag subSeqLen: Integer for the new subSeqLen value

Returns:

```
timeSeriesPredictorAPI(inputWidth, labelWidth, shift, batchSize,  
hiddenUnits, targetObject, targetFields, plotColumn, maxEpochs,  
pdf=None, removeConstantFeatures=True, optimizer='Adam',  
lstm_activation='tanh', dense_activation='sigmoid',  
lstm_initializer='glorot_uniform', dense_initializer='glorot_uniform',  
dropout=False)
```

API to replace standard command line call (the mediaPredictionRNN class has to instantiated before calling this method).

Args:

inputWidth: Integer for the length of the input sequence to be considered for the prediction
labelWidth: Integer for the length of the output sequence expected from the prediction
shift: Integer for the shift to be considered when sliding the input window
batchSize: Integer for the size of the batch to be considered when generating data sets to be fed into the RNN

model

hiddenUnits: Integer for the number of neurons contained in each hidden layer
targetObject: Name for the target Object to be considered for the time series prediction
targetFields: List of strings for the names of the object's fields to be considered for the time

series prediction

plotColumn: String for the name of the target field to be considered for plotting the input and predictions
maxEpochs: Integer for the maximum number of epochs to be considered when training the model
pdf: pdf file name to save.
removeConstantFeatures: Remove features from dict which are non-changing.
optimizer: optimizer to be used for the model, must be selected from the following

`['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',
'Adamax']`

`lstm_activation`: activation function for LSTM layers. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',
'elu', 'exponential']`

`dense_activation`: activation function for dense layer. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu',
'elu', 'exponential']`

`lstm_initializer`: weight initializer for the lstm layers. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal',
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',
'orthogonal', 'constant', 'variance_scaling']`

`dense_initializer`: weight initializer for the dense layer. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal',
'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity',
'orthogonal', 'constant', 'variance_scaling']`

`dropout`: Boolean flag to indicate if dropout should be applied in between layers in the model
`removeConstantFeatures`: Remove features from dict which are non-changing.

Returns:

`timeSeriesPredictorAllAPI(matrixProfileFlag=False,
embeddedEncodingFlag=False, categoricalEncodingFlag=False,
inputWidth=70, labelWidth=20, shift=2, batchSize=32,
hiddenUnits=128, maxEpochs=2096, subSequenceLength=1,
targetObject=None, targetFields=None, plotColumn=None, pdf=None,
currDataDict=None, debug=False, optimizer='Adam',
lstm_activation='tanh', dense_activation='sigmoid',`

```
lstm_initializer='glorot_uniform', dense_initializer='glorot_uniform',  
dropout=False, removeConstantFeatures=False)
```

API to replace standard command line call (the mediaPredictionRNN class has to instantiated before calling this method). Args:

matrixProfileFlag: Boolean value for matrixProfileFlag
embeddedEncodingFlag: Boolean value for the embeddedEncoding flag categoricalEncodingFlag: Boolean value for the categoricalEncoding flag inputWidth: Integer for the length of the input sequence to be considered for the prediction labelWidth: Integer for the length of the output sequence expected from the prediction shift: Integer for the shift to be considered when sliding the input window
batchSize: Integer for the size of the batch to be considered when generating data sets to be fed into the RNN model
hiddenUnits: Integer for the number of neurons contained in each hidden layer maxEpochs: Integer for the maximum number of epochs to be considered when training the model
subSequenceLength: Integer for the new subSeqLen value
targetObject: Name for the target Object to be considered for the time series prediction targetFields: List of strings for the names of the object's fields to be considered for the time series prediction plotColumn: String for the name of the target field to be considered for plotting the input and predictions pdf: pdf file name to save. currDataDict: dictionary containing flattened structures to be fed into RNN. Use None when not feeding entire telemetry payload into RNN debug: developer debug flag optimizer: optimizer to be used for the model, must be selected from the following

```
['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam',  
'Adamax']
```

lstm_activation: activation function for LSTM layers. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']`

`dense_activation`: activation function for dense layer. Must be selected from the following:

`['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']`

`lstm_initializer`: weight initializer for the lstm layers. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']`

`dense_initializer`: weight initializer for the dense layer. Must be selected from the following:

`['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']`

`dropout`: Boolean flag to indicate if dropout should be applied in between layers in the model.

`removeConstantFeatures`: Remove features from dict which are non-changing.

Returns:

`writeRNNtoPDF(inputWidth, labelWidth, shift, batchSize, hiddenUnits,
targetObject, targetFields, plotColumn, maxEpochs,
pdfFileName=None)`

`mediaErrorPredictor.py`

This module contains the basic functions for loading the content of a configuration file and utilizing the ARMA model to generate predictions on the time series values passed in as arguments into the main API

Args:

--inputFile: String representation for the name of the configuration file where the time series data values for the media are contained

--targetObject: String representation for the name of the target object to be used for the prediction model --targetField: String representation of the name for the target field to be used for the prediction model --matrixProfile: Boolean flag to apply matrix profile to time series before using the ARMA model --subSeqLen: Integer for the length of the sliding window for matrix profile (only relevant if matrix profile flag is set)

--debug: Boolean flag to activate verbose printing for debug use

Example:

Default usage:

```
$ python mediaErrorPredictor.py
```

Specific usage:

```
$ python mediaErrorPredictor.py --inputFile time-series.ini --targetObject NandStats --targetField 01-biterrors --matrixProfile True --subSeqLen 20 --debug True
```

class

```
src.software.MEP.mediaErrorPredictor.MediaErrorPredictor(inputFile,  
matrixProfile=False, subSeqLen=20, debug=False)
```

Bases: **object**

Class for media failure prediction

Attributes:

inputFile: String representation for the name of the configuration file where all the data values are contained

debug: Boolean flag to activate debug statements *dataDict*: Dictionary with the object values from the ConfigParser

ARMAModel(*currentObject*, *currentField*, *pdf=None*)

function for using the ARMA model on a specified time-series and producing a graph comparing predicted values with expected values

Args:

pdf: file to save to PDF format. currentObject: String representation for the name of the object to be processed
currentField: String representation for the name of the object's field to be processed

Returns:

check_stationarity(*timeseriesCandidate*)

Augmented Dickey-Fuller (ADF) test can be used to test the null hypothesis that the series is non-stationary. The ADF test helps to understand whether a change in Y is a linear trend or not. If there is a linear trend but the lagged value cannot explain the change in Y over time, then our data will be deemed non-stationary. The value of test statistics is less than 5% critical value and p-value is also less than 0.05 so we can reject the null hypothesis and Alternate Hypothesis that time series is Stationary seems to be true. When there is nothing unusual about the time plot and there appears to be no need to do any data adjustments. There is no evidence of changing variance also so we will not do a Box-Cox transformation.

Args:

timeseriesCandidate: Time series

Returns: Determination of whether the data is stationary or not?

predictorAPI(*targetObject*, *targetField*)

API to replace standard command line call (the MediaErrorPredictor class has to instantiated before calling this method).

Args:

targetObject: String representation for the name of the object to be processed targetField: String representation for the name of

the object's field to be processed

Returns:

`setMatrixProfileFlag(value, subSeqLen=20)`

`writeARMAToPDF(targetObject, targetField, outFile=None)`

Args:

`outFile`: `targetObject`: `targetField`:

Returns:

`src.software.MEP.mediaErrorPredictor.addValue_labels(ax, spacing=5)`

Add labels to the end of each bar in a bar chart.

Arguments:

`ax` (`matplotlib.axes.Axes`): The matplotlib object containing the axes of the plot to annotate.

`spacing` (int): The distance between the labels and the bars.

`src.software.MEP.mediaErrorPredictor.main()`

main function to be called when the script is directly executed from the command line

`mediaErrorGUI.py`

this module contains the basic function for generating a window-based GUI for mediaErrorPredictor

Example:

Default Usage:

`$ python mediaErrorGUI.py`

`class src.software.MEP.mediaErrorGUI.Fields(value)`

Bases: `enum.Enum`

An enumeration.

TRACKING_VARS = 0

class src.software.MEP.mediaErrorGUI.FieldsAttributes

Bases: `object`

FieldsLabels = {*Fields.TRACKING_VARS*: 'Select tracking variable'}

FieldsLegendLocation = {*Fields.TRACKING_VARS*: (0, 1.04)}

FieldsSelectionMode = {*Fields.TRACKING_VARS*: 'single'}

class

src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow(**args*, ***kwargs*)

Bases: `tkinter.Tk`

Initialization function that initializes the Tk framework

Args:

`*args`(list): The first parameter containing the variable list of arguments
`**kwargs`(dict): The second parameter contained named variable arguments.

*after(ms, func=None, *args)*

Call function once after given time.

MS specifies the time in milliseconds. FUNC gives the function which shall be called. Additional parameters are given as parameters to the function call. Return identifier to cancel scheduling with `after_cancel`.

after_cancel(id)

Cancel scheduling of function identified with ID.

Identifier returned by `after` or `after_idle` must be given as first parameter.

`after_idle(func, *args)`

Call FUNC once if the Tcl main loop has no event to process.

Return an identifier to cancel the scheduling with `after_cancel`.

`anchor(anchor=None)`

The anchor value controls how to place the grid within the master when no row/column has any weight.

The default anchor is nw.

`aspect(minNumer=None, minDenom=None, maxNumer=None, maxDenom=None)`

Instruct the window manager to set the aspect ratio (width/height) of this widget to be between MINNUMER/MINDENOM and MAXNUMBER/MAXDENOM. Return a tuple of the actual values if no argument is given.

`attributes(*args)`

This subcommand returns or sets platform specific attributes

The first form returns a list of the platform specific flags and their values. The second form returns the value for the specific option. The third form sets one or more of the values. The values are as follows:

On Windows, -disabled gets or sets whether the window is in a disabled state. -toolwindow gets or sets the style of the window to toolwindow (as defined in the MSDN). -topmost gets or sets whether this is a topmost window (displays above all other windows).

On Macintosh, XXXXX

On Unix, there are currently no special attribute values.

`bbox(column=None, row=None, col2=None, row2=None)`

Return a tuple of integer coordinates for the bounding box of this widget controlled by the geometry manager grid.

If COLUMN, ROW is given the bounding box applies from the cell with row and column 0 to the specified cell. If COL2 and ROW2 are given the bounding box starts at that cell.

The returned integers specify the offset of the upper left corner in the master widget and the width and height.

bell(*displayof*=0)

Ring a display's bell.

bind(*sequence*=None, *func*=None, *add*=None)

Bind to this widget at event SEQUENCE a call to function FUNC.

SEQUENCE is a string of concatenated event patterns. An event pattern is of the form <MODIFIER-MODIFIER-TYPE-DETAIL> where MODIFIER is one of Control, Mod2, M2, Shift, Mod3, M3, Lock, Mod4, M4, Button1, B1, Mod5, M5 Button2, B2, Meta, M, Button3, B3, Alt, Button4, B4, Double, Button5, B5 Triple, Mod1, M1. TYPE is one of Activate, Enter, Map, ButtonPress, Button, Expose, Motion, ButtonRelease FocusIn, MouseWheel, Circulate, FocusOut, Property, Colormap, Gravity Reparent, Configure, KeyPress, Key, Unmap, Deactivate, KeyRelease Visibility, Destroy, Leave and DETAIL is the button number for ButtonPress, ButtonRelease and DETAIL is the Keysym for KeyPress and KeyRelease. Examples are <Control-Button-1> for pressing Control and mouse button 1 or <Alt-A> for pressing A and the Alt key (KeyPress can be omitted). An event pattern can also be a virtual event of the form <<AString>> where AString can be arbitrary. This event can be generated by event_generate. If events are concatenated they must appear shortly after each other.

FUNC will be called if the event sequence occurs with an instance of Event as argument. If the return value of FUNC is "break" no further bound function is invoked.

An additional boolean parameter ADD specifies whether FUNC will be called additionally to the other bound function or whether it will replace the previous function.

Bind will return an identifier to allow deletion of the bound function with unbind without memory leak.

If FUNC or SEQUENCE is omitted the bound function or list of bound events are returned.

`bind_all(sequence=None, func=None, add=None)`

Bind to all widgets at an event SEQUENCE a call to function FUNC. An additional boolean parameter ADD specifies whether FUNC will be called additionally to the other bound function or whether it will replace the previous function. See bind for the return value.

`bind_class(className, sequence=None, func=None, add=None)`

Bind to widgets with bindtag CLASSNAME at event SEQUENCE a call of function FUNC. An additional boolean parameter ADD specifies whether FUNC will be called additionally to the other bound function or whether it will replace the previous function. See bind for the return value.

`bindtags(tagList=None)`

Set or get the list of bindtags for this widget.

With no argument return the list of all bindtags associated with this widget. With a list of strings as argument the bindtags are set to this list. The bindtags determine in which order events are processed (see bind).

`cget(key)`

Return the resource value for a KEY given as string.

`client(name=None)`

Store NAME in WM_CLIENT_MACHINE property of this widget.
Return current value.

`clipboard_append(string, **kw)`

Append STRING to the Tk clipboard.

A widget specified at the optional displayof keyword argument specifies the target display. The clipboard can be retrieved with selection_get.

`clipboard_clear(**kw)`

Clear the data in the Tk clipboard.

A widget specified for the optional displayof keyword argument specifies the target display.

`clipboard_get(**kw)`

Retrieve data from the clipboard on window's display.

The window keyword defaults to the root window of the Tkinter application.

The type keyword specifies the form in which the data is to be returned and should be an atom name such as STRING or FILE_NAME. Type defaults to STRING, except on X11, where the default is to try UTF8_STRING and fall back to STRING.

This command is equivalent to:

`selection_get(CLIPBOARD)`

`colormapwindows(*wlist)`

Store list of window names (WLIST) into WM_COLORMAPWINDOWS property of this widget. This list contains windows whose colormaps differ from their parents. Return current list of widgets if WLIST is empty.

`columnconfigure(index, cnf={}, **kw)`

Configure column INDEX of a grid.

Valid resources are minsize (minimum size of the column), weight (how much does additional space propagate to this column) and pad (how much space to let additionally).

`command(value=None)`

Store VALUE in WM_COMMAND property. It is the command which shall be used to invoke the application. Return current command if VALUE is None.

`config(cnf=None, **kw)`

Configure resources of a widget.

The values for resources are specified as keyword arguments. To get an overview about the allowed keyword arguments call the method keys.

`configure(cnf=None, **kw)`

Configure resources of a widget.

The values for resources are specified as keyword arguments. To get an overview about the allowed keyword arguments call the method keys.

`deiconify()`

Deiconify this widget. If it was never mapped it will not be mapped. On Windows it will raise this widget and give it the focus.

`deletecommand(name)`

Internal function.

Delete the Tcl command provided in NAME.

`destroy()`

Destroy this and all descendants widgets. This will end the application of this Tcl interpreter.

`event_add(virtual, *sequences)`

Bind a virtual event VIRTUAL (of the form <<Name>>) to an event SEQUENCE such that the virtual event is triggered whenever SEQUENCE occurs.

`event_delete(virtual, *sequences)`

Unbind a virtual event VIRTUAL from SEQUENCE.

`event_generate(sequence, **kw)`

Generate an event SEQUENCE. Additional keyword arguments specify parameter of the event (e.g. x, y, rootx, rooty).

`event_info(virtual=None)`

Return a list of all virtual events or the information about the SEQUENCE bound to the virtual event VIRTUAL.

`focus()`

Direct input focus to this widget.

If the application currently does not have the focus this widget will get the focus if the application gets the focus through the window manager.

`focus_displayof()`

Return the widget which has currently the focus on the display where this widget is located.

Return None if the application does not have the focus.

`focus_force()`

Direct input focus to this widget even if the application does not have the focus. Use with caution!

`focus_get()`

Return the widget which has currently the focus in the application.

Use `focus_displayof` to allow working with several displays. Return `None` if application does not have the focus.

`focus_lastfor()`

Return the widget which would have the focus if top level for this widget gets the focus from the window manager.

`focus_set()`

Direct input focus to this widget.

If the application currently does not have the focus this widget will get the focus if the application gets the focus through the window manager.

`focusmodel(model=None)`

Set focus model to MODEL. "active" means that this widget will claim the focus itself, "passive" means that the window manager shall give the focus. Return current focus model if MODEL is None.

`forget(window)`

The window will be unmapped from the screen and will no longer be managed by wm. toplevel windows will be treated like frame windows once they are no longer managed by wm, however, the menu option configuration will be remembered and the menus will return once the widget is managed again.

`frame()`

Return identifier for decorative frame of this widget if present.

`geometry(newGeometry=None)`

Set geometry to NEWGEOMETRY of the form =widthxheight+x+y.
Return current value if `None` is given.

`getboolean(s)`

Return a boolean value for Tcl boolean values true and false given as parameter.

`getdouble(s)`

`getInt(s)`

`getvar(name='PY_VAR')`

Return value of Tcl variable NAME.

`grab_current()`

Return widget which has currently the grab in this application or None.

`grab_release()`

Release grab for this widget if currently set.

`grab_set()`

Set grab for this widget.

A grab directs all events to this and descendant widgets in the application.

`grab_set_global()`

Set global grab for this widget.

A global grab directs all events to this and descendant widgets on the display. Use with caution - other applications do not get events anymore.

`grab_status()`

Return None, "local" or "global" if this widget has no, a local or a global grab.

`grid(baseWidth=None, baseHeight=None, widthInc=None, heightInc=None)`

Instruct the window manager that this widget shall only be resized on grid boundaries. WIDTHINC and HEIGHTINC are the width and height of a grid unit in pixels. BASEWIDTH and BASEHEIGHT are the number of grid units requested in Tk_GeometryRequest.

`grid_anchor(anchor=None)`

The anchor value controls how to place the grid within the master when no row/column has any weight.

The default anchor is nw.

`grid_bbox(column=None, row=None, col2=None, row2=None)`

Return a tuple of integer coordinates for the bounding box of this widget controlled by the geometry manager grid.

If COLUMN, ROW is given the bounding box applies from the cell with row and column 0 to the specified cell. If COL2 and ROW2 are given the bounding box starts at that cell.

The returned integers specify the offset of the upper left corner in the master widget and the width and height.

`grid_columnconfigure(index, cnf={}, **kw)`

Configure column INDEX of a grid.

Valid resources are minsize (minimum size of the column), weight (how much does additional space propagate to this column) and pad (how much space to let additionally).

`grid_location(x, y)`

Return a tuple of column and row which identify the cell at which the pixel at position X and Y inside the master widget is located.

`grid_propagate(flag=['_noarg_'])`

Set or get the status for propagation of geometry information.

A boolean argument specifies whether the geometry information of the slaves will determine the size of this widget. If no argument is given, the current setting will be returned.

`grid_rowconfigure(index, cnf={}, **kw)`

Configure row INDEX of a grid.

Valid resources are minsize (minimum size of the row), weight (how much does additional space propagate to this row) and pad (how much space to let additionally).

`grid_size()`

Return a tuple of the number of column and rows in the grid.

`grid_slaves(row=None, column=None)`

Return a list of all slaves of this widget in its packing order.

`group(pathName=None)`

Set the group leader widgets for related widgets to PATHNAME.

Return the group leader of this widget if None is given.

`iconbitmap(bitmap=None, default=None)`

Set bitmap for the iconified widget to BITMAP. Return the bitmap if None is given.

Under Windows, the DEFAULT parameter can be used to set the icon for the widget and any descendants that don't have an icon set explicitly. DEFAULT can be the relative path to a .ico file (example: root.iconbitmap(default='myicon.ico')). See Tk documentation for more information.

`iconify()`

Display widget as icon.

`iconmask(bitmap=None)`

Set mask for the icon bitmap of this widget. Return the mask if None is given.

`iconname(newName=None)`

Set the name of the icon for this widget. Return the name if None is given.

`iconphoto(default=False, *args)`

Sets the titlebar icon for this window based on the named photo images passed through args. If default is True, this is applied to all future created toplevels as well.

The data in the images is taken as a snapshot at the time of invocation. If the images are later changed, this is not reflected to the titlebar icons. Multiple images are accepted to allow different images sizes to be provided. The window manager may scale provided icons to an appropriate size.

On Windows, the images are packed into a Windows icon structure. This will override an icon specified to `wm_iconbitmap`, and vice versa.

On X, the images are arranged into the `_NET_WM_ICON_X` property, which most modern window managers support. An icon specified by `wm_iconbitmap` may exist simultaneously.

On Macintosh, this currently does nothing.

`iconposition(x=None, y=None)`

Set the position of the icon of this widget to X and Y. Return a tuple of the current values of X and Y if None is given.

`iconwindow(pathName=None)`

Set widget PATHNAME to be displayed instead of icon. Return the current value if None is given.

`image_names()`

Return a list of all existing image names.

`image_types()`

Return a list of all available image types (e.g. photo bitmap).

`init_frame()`

`keys()`

Return a list of all resource names of this widget.

`lift(aboveThis=None)`

Raise this widget in the stacking order.

`loadtk()`

`lower(belowThis=None)`

Lower this widget in the stacking order.

`mainloop(n=0)`

Call the mainloop of Tk.

`manage(widget)`

The widget specified will become a stand alone top-level window. The window will be decorated with the window managers title bar, etc.

`maxsize(width=None, height=None)`

Set max WIDTH and HEIGHT for this widget. If the window is gridded the values are given in grid units. Return the current values if None is given.

`minsize(width=None, height=None)`

Set min WIDTH and HEIGHT for this widget. If the window is gridded the values are given in grid units. Return the current values if None is given.

`nametowidget(name)`

Return the Tkinter instance of a widget identified by its Tcl name NAME.

`option_add(pattern, value, priority=None)`

Set a VALUE (second parameter) for an option PATTERN (first parameter).

An optional third parameter gives the numeric priority (defaults to 80).

`option_clear()`

Clear the option database.

It will be reloaded if option_add is called.

`option_get(name, className)`

Return the value for an option NAME for this widget with CLASSNAME.

Values with higher priority override lower values.

`option_readfile(fileName, priority=None)`

Read file FILENAME into the option database.

An optional second parameter gives the numeric priority.

`overrideredirect(boolean=None)`

Instruct the window manager to ignore this widget if BOOLEAN is given with 1. Return the current value if None is given.

`pack_propagate(flag=['noarg_'])`

Set or get the status for propagation of geometry information.

A boolean argument specifies whether the geometry information of the slaves will determine the size of this widget. If no argument is

given the current setting will be returned.

`pack_slaves()`

Return a list of all slaves of this widget in its packing order.

`place_slaves()`

Return a list of all slaves of this widget in its packing order.

`positionfrom(who=None)`

Instruct the window manager that the position of this widget shall be defined by the user if WHO is "user", and by its own policy if WHO is "program".

`propagate(flag=['noarg_'])`

Set or get the status for propagation of geometry information.

A boolean argument specifies whether the geometry information of the slaves will determine the size of this widget. If no argument is given the current setting will be returned.

`protocol(name=None, func=None)`

Bind function FUNC to command NAME for this widget. Return the function bound to NAME if None is given. NAME could be e.g. "WM_SAVE_YOURSELF" or "WM_DELETE_WINDOW".

`quit()`

Quit the Tcl interpreter. All widgets will be destroyed.

`readprofile(baseName, className)`

Internal function. It reads BASENAME.tcl and CLASSNAME.tcl into the Tcl Interpreter and calls exec on the contents of BASENAME.py and CLASSNAME.py if such a file exists in the home directory.

`register(func, subst=None, needcleanup=1)`

Return a newly created Tcl function. If this function is called, the Python function FUNC will be executed. An optional function SUBST can be given which will be executed before FUNC.

`report_callback_exception(exc, val, tb)`

Report callback exception on sys.stderr.

Applications may want to override this internal function, and should when sys.stderr is None.

`resizable(width=None, height=None)`

Instruct the window manager whether this width can be resized in WIDTH or HEIGHT. Both values are boolean values.

`rowconfigure(index, cnf={}, **kw)`

Configure row INDEX of a grid.

Valid resources are minsize (minimum size of the row), weight (how much does additional space propagate to this row) and pad (how much space to let additionally).

`selection_clear(**kw)`

Clear the current X selection.

`selection_get(**kw)`

Return the contents of the current X selection.

A keyword parameter selection specifies the name of the selection and defaults to PRIMARY. A keyword parameter displayof specifies a widget on the display to use. A keyword parameter type specifies the form of data to be fetched, defaulting to STRING except on X11, where UTF8_STRING is tried before STRING.

`selection_handle(command, **kw)`

Specify a function COMMAND to call if the X selection owned by this widget is queried by another application.

This function must return the contents of the selection. The function will be called with the arguments OFFSET and LENGTH which allows the chunking of very long selections. The following keyword parameters can be provided: selection - name of the selection (default PRIMARY), type - type of the selection (e.g. STRING, FILE_NAME).

`selection_own(**kw)`

Become owner of X selection.

A keyword parameter selection specifies the name of the selection (default PRIMARY).

`selection_own_get(**kw)`

Return owner of X selection.

The following keyword parameter can be provided: selection - name of the selection (default PRIMARY), type - type of the selection (e.g. STRING, FILE_NAME).

`send(interp, cmd, *args)`

Send Tcl command CMD to different interpreter INTERP to be executed.

`setvar(name='PY_VAR', value='I')`

Set Tcl variable NAME to VALUE.

`show_frame()`

`size()`

Return a tuple of the number of column and rows in the grid.

`sizefrom(who=None)`

Instruct the window manager that the size of this widget shall be defined by the user if WHO is "user", and by its own policy if WHO is "program".

`slaves()`

Return a list of all slaves of this widget in its packing order.

`state(newstate=None)`

Query or set the state of this widget as one of normal, icon, iconic (see `wm_iconwindow`), withdrawn, or zoomed (Windows only).

`title(string=None)`

Set the title of this widget.

`tk_bisque()`

Change the color scheme to light brown as used in Tk 3.6 and before.

`tk_focusFollowsMouse()`

The widget under mouse will get automatically focus. Can not be disabled easily.

`tk_focusNext()`

Return the next widget in the focus order which follows widget which has currently the focus.

The focus order first goes to the next child, then to the children of the child recursively and then to the next sibling which is higher in the stacking order. A widget is omitted if it has the `takefocus` resource set to 0.

`tk_focusPrev()`

Return previous widget in the focus order. See `tk_focusNext` for details.

`tk_setPalette(*args, **kw)`

Set a new color scheme for all widget elements.

A single color as argument will cause that all colors of Tk widget elements are derived from this. Alternatively several keyword

parameters and its associated colors can be given. The following keywords are valid: activeBackground, foreground, selectColor, activeForeground, highlightBackground, selectBackground, background, highlightColor, selectForeground, disabledForeground, insertBackground, troughColor.

`tk_strictMotif(boolean=None)`

Set Tcl internal variable, whether the look and feel should adhere to Motif.

A parameter of 1 means adhere to Motif (e.g. no color change if mouse passes over slider). Returns the set value.

`tkraise(aboveThis=None)`

Raise this widget in the stacking order.

`transient(master=None)`

Instruct the window manager that this widget is transient with regard to widget MASTER.

`unbind(sequence, funcid=None)`

Bind for this widget for event SEQUENCE the function identified with FUNCID.

`unbind_all(sequence)`

Bind for all widgets for event SEQUENCE all functions.

`unbind_class(className, sequence)`

Bind for all widgets with bindtag CLASSNAME for event SEQUENCE all functions.

`update()`

Enter event loop until all pending events have been processed by Tcl.

`update_idletasks()`

Enter event loop until all idle callbacks have been called. This will update the display of windows but not process events caused by the user.

`wait_variable(name='PY_VAR')`

Wait until the variable is modified.

A parameter of type IntVar, StringVar, DoubleVar or BooleanVar must be given.

`wait_visibility(window=None)`

Wait until the visibility of a WIDGET changes (e.g. it appears).

If no parameter is given self is used.

`wait_window(window=None)`

Wait until a WIDGET is destroyed.

If no parameter is given self is used.

`waitvar(name='PY_VAR')`

Wait until the variable is modified.

A parameter of type IntVar, StringVar, DoubleVar or BooleanVar must be given.

`winfo_atom(name, displayof=0)`

Return integer which represents atom NAME.

`winfo_atomname(id, displayof=0)`

Return name of atom with identifier ID.

`winfo_cells()`

Return number of cells in the colormap for this widget.

`winfo_children()`

Return a list of all widgets which are children of this widget.

winfo_class()

Return window class name of this widget.

winfo_colormapfull()

Return True if at the last color request the colormap was full.

winfo_containing(*rootX*, *rootY*, *displayof*=0)

Return the widget which is at the root coordinates ROOTX, ROOTY.

winfo_depth()

Return the number of bits per pixel.

winfo_exists()

Return true if this widget exists.

winfo_fpixels(*number*)

Return the number of pixels for the given distance NUMBER (e.g. "3c") as float.

winfo_geometry()

Return geometry string for this widget in the form "widthxheight+X+Y".

winfo_height()

Return height of this widget.

winfo_id()

Return identifier ID for this widget.

winfo_interps(*displayof*=0)

Return the name of all Tcl interpreters for this display.

`winfo_ismapped()`

Return true if this widget is mapped.

`winfo_manager()`

Return the window manager name for this widget.

`winfo_name()`

Return the name of this widget.

`winfo_parent()`

Return the name of the parent of this widget.

`winfo_pathname(id, displayof=0)`

Return the pathname of the widget given by ID.

`winfo_pixels(number)`

Rounded integer value of `winfo_fpixels`.

`winfo_pointerx()`

Return the x coordinate of the pointer on the root window.

`winfo_pointery()`

Return a tuple of x and y coordinates of the pointer on the root window.

`winfo_pointery()`

Return the y coordinate of the pointer on the root window.

`winfo_reqheight()`

Return requested height of this widget.

`winfo_reqwidth()`

Return requested width of this widget.

`winfo_rgb(color)`

Return a tuple of integer RGB values in range(65536) for color in this widget.

winfo_rootx()

Return x coordinate of upper left corner of this widget on the root window.

winfo_rooty()

Return y coordinate of upper left corner of this widget on the root window.

winfo_screen()

Return the screen name of this widget.

winfo_screencells()

Return the number of the cells in the colormap of the screen of this widget.

winfo_screendepth()

Return the number of bits per pixel of the root window of the screen of this widget.

winfo_screenheight()

Return the number of pixels of the height of the screen of this widget in pixel.

winfo_screenmmheight()

Return the number of pixels of the height of the screen of this widget in mm.

winfo_screenmmwidth()

Return the number of pixels of the width of the screen of this widget in mm.

winfo_screenvisual()

Return one of the strings directcolor, grayscale, pseudocolor, staticcolor, staticgray, or truecolor for the default colormodel of this screen.

winfo_screenwidth()

Return the number of pixels of the width of the screen of this widget in pixel.

winfo_server()

Return information of the X-Server of the screen of this widget in the form "XmajorRminor vendor vendorVersion".

winfo_toplevel()

Return the toplevel widget of this widget.

winfo_viewable()

Return true if the widget and all its higher ancestors are mapped.

winfo_visual()

Return one of the strings directcolor, grayscale, pseudocolor, staticcolor, staticgray, or truecolor for the colormodel of this widget.

winfo_visualid()

Return the X identifier for the visual for this widget.

winfo_visualsavailable(*includeids=False*)

Return a list of all visuals available for the screen of this widget.

Each item in the list consists of a visual name (see winfo_visual), a depth and if includeids is true is given also the X identifier.

winfo_vrootheight()

Return the height of the virtual root window associated with this widget in pixels. If there is no virtual root window return the height of the screen.

winfo_vrootwidth()

Return the width of the virtual root window associated with this widget in pixel. If there is no virtual root window return the width of the screen.

winfo_vrootx()

Return the x offset of the virtual root relative to the root window of the screen of this widget.

winfo_vrooty()

Return the y offset of the virtual root relative to the root window of the screen of this widget.

winfo_width()

Return the width of this widget.

winfo_x()

Return the x coordinate of the upper left corner of this widget in the parent.

winfo_y()

Return the y coordinate of the upper left corner of this widget in the parent.

withdraw()

Withdraw this widget from the screen such that it is unmapped and forgotten by the window manager. Re-draw it with `wm_deiconify`.

wm_aspect(*minNumer=None*, *minDenom=None*, *maxNumer=None*, *maxDenom=None*)

Instruct the window manager to set the aspect ratio (width/height) of this widget to be between MINNUMER/MINDENOM and MAXNUMER/MAXDENOM. Return a tuple of the actual values if no argument is given.

wm_attributes(*args)

This subcommand returns or sets platform specific attributes

The first form returns a list of the platform specific flags and their values. The second form returns the value for the specific option. The third form sets one or more of the values. The values are as follows:

On Windows, -disabled gets or sets whether the window is in a disabled state. -toolwindow gets or sets the style of the window to toolwindow (as defined in the MSDN). -topmost gets or sets whether this is a topmost window (displays above all other windows).

On Macintosh, XXXXX

On Unix, there are currently no special attribute values.

wm_client(*name=None*)

Store NAME in WM_CLIENT_MACHINE property of this widget.
Return current value.

wm_colormapwindows(*wlist)

Store list of window names (WLIST) into
WM_COLORMAPWINDOWS property of this widget. This list
contains windows whose colormaps differ from their parents. Return
current list of widgets if WLIST is empty.

wm_command(*value=None*)

Store VALUE in WM_COMMAND property. It is the command
which shall be used to invoke the application. Return current
command if VALUE is None.

wm_deiconify()

Deiconify this widget. If it was never mapped it will not be mapped.
On Windows it will raise this widget and give it the focus.

`wm_focusmodel(model=None)`

Set focus model to MODEL. "active" means that this widget will claim the focus itself, "passive" means that the window manager shall give the focus. Return current focus model if MODEL is None.

`wm_forget(window)`

The window will be unmapped from the screen and will no longer be managed by wm. toplevel windows will be treated like frame windows once they are no longer managed by wm, however, the menu option configuration will be remembered and the menus will return once the widget is managed again.

`wm_frame()`

Return identifier for decorative frame of this widget if present.

`wm_geometry(newGeometry=None)`

Set geometry to NEWGEOMETRY of the form =widthxheight+x+y.
Return current value if None is given.

`wm_grid(baseWidth=None, baseHeight=None, widthInc=None,
heightInc=None)`

Instruct the window manager that this widget shall only be resized on grid boundaries. WIDTHINC and HEIGHTINC are the width and height of a grid unit in pixels. BASEWIDTH and BASEHEIGHT are the number of grid units requested in Tk_GeometryRequest.

`wm_group(pathName=None)`

Set the group leader widgets for related widgets to PATHNAME.
Return the group leader of this widget if None is given.

`wm_iconbitmap(bitmap=None, default=None)`

Set bitmap for the iconified widget to BITMAP. Return the bitmap if None is given.

Under Windows, the DEFAULT parameter can be used to set the icon for the widget and any descendants that don't have an icon set explicitly. DEFAULT can be the relative path to a .ico file (example: root.iconbitmap(default='myicon.ico')). See Tk documentation for more information.

wm_iconify()

Display widget as icon.

wm_iconmask(*bitmap=None*)

Set mask for the icon bitmap of this widget. Return the mask if None is given.

wm_icongame(*newName=None*)

Set the name of the icon for this widget. Return the name if None is given.

wm_iconphoto(*default=False, *args*)

Sets the titlebar icon for this window based on the named photo images passed through args. If default is True, this is applied to all future created toplevels as well.

The data in the images is taken as a snapshot at the time of invocation. If the images are later changed, this is not reflected to the titlebar icons. Multiple images are accepted to allow different images sizes to be provided. The window manager may scale provided icons to an appropriate size.

On Windows, the images are packed into a Windows icon structure. This will override an icon specified to `wm_iconbitmap`, and vice versa.

On X, the images are arranged into the `_NET_WM_ICON_X` property, which most modern window managers support. An icon specified by `wm_iconbitmap` may exist simultaneously.

On Macintosh, this currently does nothing.

`wm_iconposition(x=None, y=None)`

Set the position of the icon of this widget to X and Y. Return a tuple of the current values of X and Y if None is given.

`wm_iconwindow(pathName=None)`

Set widget PATHNAME to be displayed instead of icon. Return the current value if None is given.

`wm_manage(widget)`

The widget specified will become a stand alone top-level window. The window will be decorated with the window managers title bar, etc.

`wm_maxsize(width=None, height=None)`

Set max WIDTH and HEIGHT for this widget. If the window is gridded the values are given in grid units. Return the current values if None is given.

`wm_minsize(width=None, height=None)`

Set min WIDTH and HEIGHT for this widget. If the window is gridded the values are given in grid units. Return the current values if None is given.

`wm_overrideredirect(boolean=None)`

Instruct the window manager to ignore this widget if BOOLEAN is given with 1. Return the current value if None is given.

`wm_positionfrom(who=None)`

Instruct the window manager that the position of this widget shall be defined by the user if WHO is "user", and by its own policy if WHO is "program".

`wm_protocol(name=None, func=None)`

Bind function FUNC to command NAME for this widget. Return the function bound to NAME if None is given. NAME could be e.g. "WM_SAVE_YOURSELF" or "WM_DELETE_WINDOW".

`wm_resizable(width=None, height=None)`

Instruct the window manager whether this width can be resized in WIDTH or HEIGHT. Both values are boolean values.

`wm_sizefrom(who=None)`

Instruct the window manager that the size of this widget shall be defined by the user if WHO is "user", and by its own policy if WHO is "program".

`wm_state(newstate=None)`

Query or set the state of this widget as one of normal, icon, iconic (see `wm_iconwindow`), withdrawn, or zoomed (Windows only).

`wm_title(string=None)`

Set the title of this widget.

`wm_transient(master=None)`

Instruct the window manager that this widget is transient with regard to widget MASTER.

`wm_withdraw()`

Withdraw this widget from the screen such that it is unmapped and forgotten by the window manager. Re-draw it with `wm_deiconify`.

`class src.software.MEP.mediaErrorGUI.ObjectConfig(configFilePath)`

Bases: `object`

`objectFilePath = None`

`objectIDs = []`

`readConfigContent(debug=False)`

`trackingVars = []`

`class src.software.MEP.mediaErrorGUI.StartPage(parent, controller)`

Bases: `tkinter.Frame`

after(ms, func=None, *args)

Call function once after given time.

MS specifies the time in milliseconds. FUNC gives the function which shall be called. Additional parameters are given as parameters to the function call. Return identifier to cancel scheduling with `after_cancel`.

after_cancel(id)

Cancel scheduling of function identified with ID.

Identifier returned by `after` or `after_idle` must be given as first parameter.

after_idle(func, *args)

Call FUNC once if the Tcl main loop has no event to process.

Return an identifier to cancel the scheduling with `after_cancel`.

anchor(anchor=None)

The anchor value controls how to place the grid within the master when no row/column has any weight.

The default anchor is nw.

bbox(column=None, row=None, col2=None, row2=None)

Return a tuple of integer coordinates for the bounding box of this widget controlled by the geometry manager grid.

If COLUMN, ROW is given the bounding box applies from the cell with row and column 0 to the specified cell. If COL2 and ROW2 are given the bounding box starts at that cell.

The returned integers specify the offset of the upper left corner in the master widget and the width and height.

bell(displayof=0)

Ring a display's bell.

`bind(sequence=None, func=None, add=None)`

Bind to this widget at event SEQUENCE a call to function FUNC.

SEQUENCE is a string of concatenated event patterns. An event pattern is of the form <MODIFIER-MODIFIER-TYPE-DETAIL> where MODIFIER is one of Control, Mod2, M2, Shift, Mod3, M3, Lock, Mod4, M4, Button1, B1, Mod5, M5 Button2, B2, Meta, M, Button3, B3, Alt, Button4, B4, Double, Button5, B5 Triple, Mod1, M1. TYPE is one of Activate, Enter, Map, ButtonPress, Button, Expose, Motion, ButtonRelease FocusIn, MouseWheel, Circulate, FocusOut, Property, Colormap, Gravity Reparent, Configure, KeyPress, Key, Unmap, Deactivate, KeyRelease Visibility, Destroy, Leave and DETAIL is the button number for ButtonPress, ButtonRelease and DETAIL is the Keysym for KeyPress and KeyRelease. Examples are <Control-Button-1> for pressing Control and mouse button 1 or <Alt-A> for pressing A and the Alt key (KeyPress can be omitted). An event pattern can also be a virtual event of the form <<AString>> where AString can be arbitrary. This event can be generated by event_generate. If events are concatenated they must appear shortly after each other.

FUNC will be called if the event sequence occurs with an instance of Event as argument. If the return value of FUNC is "break" no further bound function is invoked.

An additional boolean parameter ADD specifies whether FUNC will be called additionally to the other bound function or whether it will replace the previous function.

Bind will return an identifier to allow deletion of the bound function with unbind without memory leak.

If FUNC or SEQUENCE is omitted the bound function or list of bound events are returned.

`bind_all(sequence=None, func=None, add=None)`

Bind to all widgets at an event SEQUENCE a call to function FUNC. An additional boolean parameter ADD specifies whether FUNC will be called additionally to the other bound function or whether it will replace the previous function. See bind for the return value.

`bind_class(className, sequence=None, func=None, add=None)`

Bind to widgets with bindtag CLASSNAME at event SEQUENCE a call of function FUNC. An additional boolean parameter ADD specifies whether FUNC will be called additionally to the other bound function or whether it will replace the previous function. See bind for the return value.

`bindtags(tagList=None)`

Set or get the list of bindtags for this widget.

With no argument return the list of all bindtags associated with this widget. With a list of strings as argument the bindtags are set to this list. The bindtags determine in which order events are processed (see bind).

`cget(key)`

Return the resource value for a KEY given as string.

`clipboard_append(string, **kw)`

Append STRING to the Tk clipboard.

A widget specified at the optional displayof keyword argument specifies the target display. The clipboard can be retrieved with selection_get.

`clipboard_clear(**kw)`

Clear the data in the Tk clipboard.

A widget specified for the optional displayof keyword argument specifies the target display.

`clipboard_get(**kw)`

Retrieve data from the clipboard on window's display.

The window keyword defaults to the root window of the Tkinter application.

The type keyword specifies the form in which the data is to be returned and should be an atom name such as STRING or FILE_NAME. Type defaults to STRING, except on X11, where the default is to try UTF8_STRING and fall back to STRING.

This command is equivalent to:

`selection_get(CLIPBOARD)`

`columnconfigure(index, cnf={}, **kw)`

Configure column INDEX of a grid.

Valid resources are minsize (minimum size of the column), weight (how much does additional space propagate to this column) and pad (how much space to let additionally).

`config(cnf=None, **kw)`

Configure resources of a widget.

The values for resources are specified as keyword arguments. To get an overview about the allowed keyword arguments call the method keys.

`configure(cnf=None, **kw)`

Configure resources of a widget.

The values for resources are specified as keyword arguments. To get an overview about the allowed keyword arguments call the method keys.

`create_new_graph_window()`

`deletecommand(name)`

Internal function.

Delete the Tcl command provided in NAME.

`destroy()`

Destroy this and all descendants widgets.

`display_dir()`

`event_add(virtual, *sequences)`

Bind a virtual event VIRTUAL (of the form <<Name>>) to an event SEQUENCE such that the virtual event is triggered whenever SEQUENCE occurs.

`event_delete(virtual, *sequences)`

Unbind a virtual event VIRTUAL from SEQUENCE.

`event_generate(sequence, **kw)`

Generate an event SEQUENCE. Additional keyword arguments specify parameter of the event (e.g. x, y, rootx, rooty).

`event_info(virtual=None)`

Return a list of all virtual events or the information about the SEQUENCE bound to the virtual event VIRTUAL.

`focus()`

Direct input focus to this widget.

If the application currently does not have the focus this widget will get the focus if the application gets the focus through the window manager.

`focus_displayof()`

Return the widget which has currently the focus on the display where this widget is located.

Return None if the application does not have the focus.

`focus_force()`

Direct input focus to this widget even if the application does not have the focus. Use with caution!

`focus_get()`

Return the widget which has currently the focus in the application.

Use `focus_displayof` to allow working with several displays. Return None if application does not have the focus.

`focus_lastfor()`

Return the widget which would have the focus if top level for this widget gets the focus from the window manager.

`focus_set()`

Direct input focus to this widget.

If the application currently does not have the focus this widget will get the focus if the application gets the focus through the window manager.

`forget()`

Unmap this widget and do not use it for the packing order.

`get_listbox_content(listbox)`

`getboolean(s)`

Return a boolean value for Tcl boolean values true and false given as parameter.

`getdouble(s)`

`getint(s)`

`getvar(name='PY_VAR')`

Return value of Tcl variable NAME.

grab_current()

Return widget which has currently the grab in this application or None.

grab_release()

Release grab for this widget if currently set.

grab_set()

Set grab for this widget.

A grab directs all events to this and descendant widgets in the application.

grab_set_global()

Set global grab for this widget.

A global grab directs all events to this and descendant widgets on the display. Use with caution - other applications do not get events anymore.

grab_status()

Return None, "local" or "global" if this widget has no, a local or a global grab.

grid(*cnf*={}, ***kw*)

Position a widget in the parent widget in a grid. Use as options:
column=number - use cell identified with given column (starting with 0)
columnspan=number - this widget will span several columns
in=master - use master to contain this widget in_=master - see 'in'
option description ipadx=amount - add internal padding in x direction
ipady=amount - add internal padding in y direction
padx=amount - add padding in x direction pady=amount - add padding in y direction
row=number - use cell identified with given

row (starting with 0) rowspan=number - this widget will span several rows sticky=NSEW - if cell is larger on which sides will this widget stick to the cell boundary

`grid_anchor(anchor=None)`

The anchor value controls how to place the grid within the master when no row/column has any weight.

The default anchor is nw.

`grid_bbox(column=None, row=None, col2=None, row2=None)`

Return a tuple of integer coordinates for the bounding box of this widget controlled by the geometry manager grid.

If COLUMN, ROW is given the bounding box applies from the cell with row and column 0 to the specified cell. If COL2 and ROW2 are given the bounding box starts at that cell.

The returned integers specify the offset of the upper left corner in the master widget and the width and height.

`grid_columnconfigure(index, cnf={}*, **kw)`

Configure column INDEX of a grid.

Valid resources are minsize (minimum size of the column), weight (how much does additional space propagate to this column) and pad (how much space to let additionally).

`grid_configure(cnf={}*, **kw)`

Position a widget in the parent widget in a grid. Use as options: column=number - use cell identified with given column (starting with 0) columnspan=number - this widget will span several columns in=master - use master to contain this widget in_=master - see 'in' option description ipadx=amount - add internal padding in x direction ipady=amount - add internal padding in y direction padx=amount - add padding in x direction pady=amount - add

padding in y direction row=number - use cell identified with given row (starting with 0) rowspan=number - this widget will span several rows sticky=NSEW - if cell is larger on which sides will this

widget stick to the cell boundary

`grid_forget()`

Unmap this widget.

`grid_info()`

Return information about the options for positioning this widget in a grid.

`grid_location(x, y)`

Return a tuple of column and row which identify the cell at which the pixel at position X and Y inside the master widget is located.

`grid_propagate(flag=['noarg'])`

Set or get the status for propagation of geometry information.

A boolean argument specifies whether the geometry information of the slaves will determine the size of this widget. If no argument is given, the current setting will be returned.

`grid_remove()`

Unmap this widget but remember the grid options.

`grid_rowconfigure(index, cnf={}, **kw)`

Configure row INDEX of a grid.

Valid resources are minsize (minimum size of the row), weight (how much does additional space propagate to this row) and pad (how much space to let additionally).

`grid_size()`

Return a tuple of the number of column and rows in the grid.

`grid_slaves(row=None, column=None)`

Return a list of all slaves of this widget in its packing order.

`image_names()`

Return a list of all existing image names.

`image_types()`

Return a list of all available image types (e.g. photo bitmap).

`info()`

Return information about the packing options for this widget.

`is_data_selected_from_fields()`

`keys()`

Return a list of all resource names of this widget.

`lift(aboveThis=None)`

Raise this widget in the stacking order.

`listBox2 = None`

`listbox = None`

`location(x, y)`

Return a tuple of column and row which identify the cell at which the pixel at position X and Y inside the master widget is located.

`lower(belowThis=None)`

Lower this widget in the stacking order.

`mainloop(n=0)`

Call the mainloop of Tk.

`nametowidget(name)`

Return the Tkinter instance of a widget identified by its Tcl name NAME.

`option_add(pattern, value, priority=None)`

Set a VALUE (second parameter) for an option PATTERN (first parameter).

An optional third parameter gives the numeric priority (defaults to 80).

`option_clear()`

Clear the option database.

It will be reloaded if `option_add` is called.

`option_get(name, className)`

Return the value for an option NAME for this widget with CLASSNAME.

Values with higher priority override lower values.

`option_readfile(fileName, priority=None)`

Read file FILENAME into the option database.

An optional second parameter gives the numeric priority.

`pack(cnfg={}, **kw)`

Pack a widget in the parent widget. Use as options: after=widget - pack it after you have packed widget anchor=NSEW (or subset) - position widget according to

given direction

before=widget - pack it before you will pack widget expand=bool - expand widget if parent size grows fill=NONE or X or Y or BOTH - fill widget if widget grows in=master - use master to contain this widget in_=master - see 'in' option description ipadx=amount - add

internal padding in x direction ipady=amount - add internal padding in y direction padx=amount - add padding in x direction
pady=amount - add padding in y direction side=TOP or BOTTOM or LEFT or RIGHT - where to add this widget.

`pack_configure(cnf={}, **kw)`

Pack a widget in the parent widget. Use as options: after=widget - pack it after you have packed widget anchor=NSEW (or subset) - position widget according to

given direction

before=widget - pack it before you will pack widget expand=bool - expand widget if parent size grows fill=NONE or X or Y or BOTH - fill widget if widget grows in=master - use master to contain this widget in_=master - see 'in' option description ipadx=amount - add internal padding in x direction ipady=amount - add internal padding in y direction padx=amount - add padding in x direction
pady=amount - add padding in y direction side=TOP or BOTTOM or LEFT or RIGHT - where to add this widget.

`pack_forget()`

Unmap this widget and do not use it for the packing order.

`pack_info()`

Return information about the packing options for this widget.

`pack_propagate(flag=['noarg_'])`

Set or get the status for propagation of geometry information.

A boolean argument specifies whether the geometry information of the slaves will determine the size of this widget. If no argument is given the current setting will be returned.

`pack_slaves()`

Return a list of all slaves of this widget in its packing order.

place(*cnf*={}, ***kw*)

Place a widget in the parent widget. Use as options: in=master - master relative to which the widget is placed in_=master - see 'in' option description x=amount - locate anchor of this widget at position x of master y=amount - locate anchor of this widget at position y of master relx=amount - locate anchor of this widget between 0.0 and 1.0

relative to width of master (1.0 is right edge)

rely=amount - locate anchor of this widget between 0.0 and 1.0
relative to height of master (1.0 is bottom edge)

anchor=NSEW (or subset) - position anchor according to given direction width=amount - width of this widget in pixel height=amount - height of this widget in pixel relwidth=amount - width of this widget between 0.0 and 1.0

relative to width of master (1.0 is the same width as the master)

relheight=amount - height of this widget between 0.0 and 1.0
relative to height of master (1.0 is the same height as the master)

bordermode="inside" or "outside" - whether to take border width of master widget into account

place_configure(*cnf*={}, ***kw*)

Place a widget in the parent widget. Use as options: in=master - master relative to which the widget is placed in_=master - see 'in' option description x=amount - locate anchor of this widget at position x of master y=amount - locate anchor of this widget at position y of master relx=amount - locate anchor of this widget between 0.0 and 1.0

relative to width of master (1.0 is right edge)

rely=amount - locate anchor of this widget between 0.0 and 1.0
relative to height of master (1.0 is bottom edge)

anchor=NSEW (or subset) - position anchor according to given direction
width=amount - width of this widget in pixel
height=amount - height of this widget in pixel relwidth=amount - width of this widget between 0.0 and 1.0

relative to width of master (1.0 is the same width as the master)

relheight=amount - height of this widget between 0.0 and 1.0
relative to height of master (1.0 is the same height as the master)

bordermode="inside" or "outside" - whether to take border width of master widget into account

place_forget()

Unmap this widget.

place_info()

Return information about the placing options for this widget.

place_slaves()

Return a list of all slaves of this widget in its packing order.

populate_object(*dirPath*)

propagate(*flag*=['_noarg_'])

Set or get the status for propagation of geometry information.

A boolean argument specifies whether the geometry information of the slaves will determine the size of this widget. If no argument is given the current setting will be returned.

quit()

Quit the Tcl interpreter. All widgets will be destroyed.

register(*func*, *subst=None*, *needcleanup=1*)

Return a newly created Tcl function. If this function is called, the Python function FUNC will be executed. An optional function

SUBST can be given which will be executed before FUNC.

`rowconfigure(index, cnf={}, **kw)`

Configure row INDEX of a grid.

Valid resources are minsize (minimum size of the row), weight (how much does additional space propagate to this row) and pad (how much space to let additionally).

`selection_clear(**kw)`

Clear the current X selection.

`selection_get(**kw)`

Return the contents of the current X selection.

A keyword parameter selection specifies the name of the selection and defaults to PRIMARY. A keyword parameter displayof specifies a widget on the display to use. A keyword parameter type specifies the form of data to be fetched, defaulting to STRING except on X11, where UTF8_STRING is tried before STRING.

`selection_handle(command, **kw)`

Specify a function COMMAND to call if the X selection owned by this widget is queried by another application.

This function must return the contents of the selection. The function will be called with the arguments OFFSET and LENGTH which allows the chunking of very long selections. The following keyword parameters can be provided: selection - name of the selection (default PRIMARY), type - type of the selection (e.g. STRING, FILE_NAME).

`selection_own(**kw)`

Become owner of X selection.

A keyword parameter selection specifies the name of the selection (default PRIMARY).

`selection_own_get(**kw)`

Return owner of X selection.

The following keyword parameter can be provided: selection - name of the selection (default PRIMARY), type - type of the selection (e.g. STRING, FILE_NAME).

`send(interp, cmd, *args)`

Send Tcl command CMD to different interpreter INTERP to be executed.

`setvar(name='PY_VAR', value='I')`

Set Tcl variable NAME to VALUE.

`size()`

Return a tuple of the number of column and rows in the grid.

`slaves()`

Return a list of all slaves of this widget in its packing order.

`tk_bisque()`

Change the color scheme to light brown as used in Tk 3.6 and before.

`tk_focusFollowsMouse()`

The widget under mouse will get automatically focus. Can not be disabled easily.

`tk_focusNext()`

Return the next widget in the focus order which follows widget which has currently the focus.

The focus order first goes to the next child, then to the children of the child recursively and then to the next sibling which is higher in the stacking order. A widget is omitted if it has the `takefocus` resource set to 0.

`tk_focusPrev()`

Return previous widget in the focus order. See `tk_focusNext` for details.

`tk_setPalette(*args, **kw)`

Set a new color scheme for all widget elements.

A single color as argument will cause that all colors of Tk widget elements are derived from this. Alternatively several keyword parameters and its associated colors can be given. The following keywords are valid: `activeBackground`, `foreground`, `selectColor`, `activeForeground`, `highlightBackground`, `selectBackground`, `background`, `highlightColor`, `selectForeground`, `disabledForeground`, `insertBackground`, `troughColor`.

`tk_strictMotif(boolean=None)`

Set Tcl internal variable, whether the look and feel should adhere to Motif.

A parameter of 1 means adhere to Motif (e.g. no color change if mouse passes over slider). Returns the set value.

`tkraise(aboveThis=None)`

Raise this widget in the stacking order.

`unbind(sequence, funcid=None)`

Unbind for this widget for event SEQUENCE the function identified with FUNCID.

`unbind_all(sequence)`

Unbind for all widgets for event SEQUENCE all functions.

`unbind_class(className, sequence)`

Unbind for all widgets with bindtag CLASSNAME for event SEQUENCE all functions.

`update()`

Enter event loop until all pending events have been processed by Tcl.

`update_idletasks()`

Enter event loop until all idle callbacks have been called. This will update the display of windows but not process events caused by the user.

`wait_variable(name='PY_VAR')`

Wait until the variable is modified.

A parameter of type IntVar, StringVar, DoubleVar or BooleanVar must be given.

`wait_visibility(window=None)`

Wait until the visibility of a WIDGET changes (e.g. it appears).

If no parameter is given self is used.

`wait_window(window=None)`

Wait until a WIDGET is destroyed.

If no parameter is given self is used.

`waitvar(name='PY_VAR')`

Wait until the variable is modified.

A parameter of type IntVar, StringVar, DoubleVar or BooleanVar must be given.

`winfo_atom(name, displayof=0)`

Return integer which represents atom NAME.

`winfo_atomname(id, displayof=0)`

Return name of atom with identifier ID.

`winfo_cells()`

Return number of cells in the colormap for this widget.

`winfo_children()`

Return a list of all widgets which are children of this widget.

`winfo_class()`

Return window class name of this widget.

`winfo_colormapfull()`

Return True if at the last color request the colormap was full.

`winfo_containing(rootX, rootY, displayof=0)`

Return the widget which is at the root coordinates ROOTX, ROOTY.

`winfo_depth()`

Return the number of bits per pixel.

`winfo_exists()`

Return true if this widget exists.

`winfo_fpixels(number)`

Return the number of pixels for the given distance NUMBER (e.g. "3c") as float.

`winfo_geometry()`

Return geometry string for this widget in the form "widthxheight+X+Y".

`winfo_height()`

Return height of this widget.

`winfo_id()`

Return identifier ID for this widget.

`winfo_interps(displayof=0)`

Return the name of all Tcl interpreters for this display.

`winfo_ismapped()`

Return true if this widget is mapped.

`winfo_manager()`

Return the window manager name for this widget.

`winfo_name()`

Return the name of this widget.

`winfo_parent()`

Return the name of the parent of this widget.

`winfo_pathname(id, displayof=0)`

Return the pathname of the widget given by ID.

`winfo_pixels(number)`

Rounded integer value of winfo_fpixels.

`winfo_pointerx()`

Return the x coordinate of the pointer on the root window.

`winfo_pointery()`

Return a tuple of x and y coordinates of the pointer on the root window.

`winfo_pointery()`

Return the y coordinate of the pointer on the root window.

`winfo_reqheight()`

Return requested height of this widget.

`winfo_reqwidth()`

Return requested width of this widget.

winfo_rgb(*color*)

Return a tuple of integer RGB values in range(65536) for color in this widget.

winfo_rootx()

Return x coordinate of upper left corner of this widget on the root window.

winfo_rooty()

Return y coordinate of upper left corner of this widget on the root window.

winfo_screen()

Return the screen name of this widget.

winfo_screencells()

Return the number of the cells in the colormap of the screen of this widget.

winfo_screendepth()

Return the number of bits per pixel of the root window of the screen of this widget.

winfo_screenheight()

Return the number of pixels of the height of the screen of this widget in pixel.

winfo_screenmmheight()

Return the number of pixels of the height of the screen of this widget in mm.

winfo_screenmmwidth()

Return the number of pixels of the width of the screen of this widget in mm.

winfo_screnvisual()

Return one of the strings directcolor, grayscale, pseudocolor, staticcolor, staticgray, or truecolor for the default colormodel of this screen.

winfo_screenwidth()

Return the number of pixels of the width of the screen of this widget in pixel.

winfo_server()

Return information of the X-Server of the screen of this widget in the form "XmajorRminor vendor vendorVersion".

winfo_toplevel()

Return the toplevel widget of this widget.

winfo_viewable()

Return true if the widget and all its higher ancestors are mapped.

winfo_visual()

Return one of the strings directcolor, grayscale, pseudocolor, staticcolor, staticgray, or truecolor for the colormodel of this widget.

winfo_visualid()

Return the X identifier for the visual for this widget.

winfo_visualsavailable(*includeids=False*)

Return a list of all visuals available for the screen of this widget.

Each item in the list consists of a visual name (see winfo_visual), a depth and if includeids is true is given also the X identifier.

winfo_vrootheight()

Return the height of the virtual root window associated with this widget in pixels. If there is no virtual root window return the height of the screen.

winfo_vrootwidth()

Return the width of the virtual root window associated with this widget in pixel. If there is no virtual root window return the width of the screen.

winfo_vrootx()

Return the x offset of the virtual root relative to the root window of the screen of this widget.

winfo_vrooty()

Return the y offset of the virtual root relative to the root window of the screen of this widget.

winfo_width()

Return the width of this widget.

winfo_x()

Return the x coordinate of the upper left corner of this widget in the parent.

winfo_y()

Return the y coordinate of the upper left corner of this widget in the parent.

loadAndProbeSystem.py

This module contains the basic functions for generating a load, preparing a drive and pulling data values from it, using test commands. The load is generated using FIO, so the input file must be a bash script (.sh file extension). The drive number of the device to be tested must be passed in both, the load configuration file and as an argument. The load configuration files must have time-based execution for proper execution of the script.

Args:

--driveNumber: String representation for the drive number from which to pull the data values --driveName: String representation for the name

of device interface to get data from --inputFile: String representation for the name of the input file where the workload configuration is stored --identifier: String representation for the name of the data set that corresponds to the data pull to be executed --iterations: String representation for the number of data points to be considered in the time series --outputDir: String representation for the name of the output directory where the text files will be stored --outFile: Output configuration file suffix (including .ini file extension) where the aggregated data will be stored --prepFlag: Boolean flag to indicate if the program should prep the drive before loading it --aggregateFlag: Boolean flag to indicate if the program should aggregate all the data into a configuration file

(.ini file extension)

--debug: Boolean flag to activate verbose printing for debug use

Example:

Default usage:

```
$ sudo python loadAndProbeDrive.py
```

Specific usage:

```
$ sudo python loadAndProbeDrive.py --driveNumber 0 --  
driveName /dev/nvme0n1 --inputFile rand-write.sh --identifier  
Tv2Hi --iterations 200 --outputDir test1 outFile test1.ini --prepFlag  
True -- aggregateFlag True --debug True
```

src.software.MEP.loadAndProbeDrive.API(*options=None*)

src.software.MEP.loadAndProbeDrive.inputsAPI(*driveNumber=0, driveName=None, inputFile=None, identifier='Tv2HiTAC', outputDir='.', parse=False, volumeLabel='perf', volumeAllocUnit='4096', volumeFS='ntfs', volumeLetter='g', partitionStyle='mbr', partitionFlag=True, prepFlag=True, debug=False*)

*class src.software.MEP.loadAndProbeDrive.loadDrive(*inputFile, driveNumber, driveName, identifier, iterations, outputDir, debug*)*

Bases: **object**

`aggregateData(fileListBandSizes, fileListBandStates, fileListNandStats,
fileListRberStats, outFile='time-series.ini')`

function for aggregating data values into a configuration file

Args:

fileListBandSizes: list of text files produced by the test command parseBandSizes
fileListBandStates: list of text files produced by the test command parseBandStates
fileListNandStats: list of text files produced by the test command parseNandStats
fileListRberStats: list of text files produced by the test command parseRberStats
outFile: String representation for the name of the output configuration file where the aggregated data will

be stored

Returns:

`collectDriveData(time=None, start=None)`

function for collecting data from the drive

Args:

time: Integer for the number of seconds that the collection process should last for start: Time stamp for the start point

Returns:

fileListBandSizes: List of plain text file names that contain the data values for BandSizes
fileListBandStates: List of plain text file names that contain the data values for BandStates
fileListNandStats: List of plain text file names that contain the data values for NandStats
fileListRberStats: List of plain text file names that contain the data values for RberStats
fileListTelemetry: List of binary file names that contain the data values for Telemetry

`loadDriveAPI(prepFlag=True, aggregateFlag=True, outFile='time-series.ini')`

API to replace standard command line call (the loadDrive class has to instantiated before calling this method). This function also checks that the input file exists and it has the correct suffix

Args:

prepFlag: Boolean flag to indicate if the program should prep
 the drive before loading it
 aggregateFlag: Boolean flag to activate data aggregation after collection
 outFile: String representation for the name of the output configuration file where the aggregated data will

 be stored

Returns:

`logCommandExecution(status, fileList, outFile, currentName)`

function for printing the status of the test command execution

Args:

status: Boolean value returned from the test command execution
 fileList: List containing all the files that were successfully generated using the test command
 outFile: String representation for the name of the output file where the results of the test command were

 stored

currentName: String representation for the name of the object that the test command is accessing

Returns:

`prepAndLoadDriveLinux(prepFlag=True, aggregateFlag=True, outFile='time-series.ini')`

Driver function for Linux. It calls on FIO to prepare the drives and execute the performance test in an independent subprocess, while the main process collects data values through test commands and telemetry

Args:

prepFlag: Boolean flag to indicate if the program should prep the drive before loading it
aggregateFlag: Boolean flag to activate data aggregation after collection
outFile: String representation for the name of the output configuration file where the aggregated data will be stored

Returns:

`prepDriveLinux()`

function for prepping the specified drive using an automatically generated FIO script

Returns:

`processTextFilesInList(fileList, resultDict, currentName)`

function for processing all text files contained in a list

Args:

fileList: list of text files produced by the test command
resultDict: Dictionary containing the fields and values of the objects specified in the text files
currentName: String representation for the name of the object under evaluation

Returns:

`class src.software.MEP.loadAndProbeDrive.loadDriveUtility`

Bases: `object`

Utility class containing helper methods

`IntelIMASCommand(tool='intelmas', operation='dump',
outputFile='telemetry_data.bin', driveNumber=1)`

function for generating a telemetry log using the intelmas tool commands

Args:

tool: String representation of the tool set to be used for the telemetry pull operation: String representation of the operation to be performed (refer to the notes below or the documentation for the tool chosen)

outputFile: String of the name for the output file driveNumber: Integer value for the drive ID number

Returns:

Notes:

IMAS

https://downloadmirror.intel.com/29337/eng/Intel_Memory_and_Storage_Tool_User%20Guide-Public-342245-001US.pdf

<https://downloadcenter.intel.com/download/29337/Intel-Memory-and-Storage-Tool-CLI-Command-Line-Interface-Commands>

```
intelmas show -o json -intelssd intelmas show -intelssd  
1 -identify intelmas dump -destination  
telemetry_data.bin -intelssd 1 -telemetrylog intelmas  
dump -destination telemetry_data.bin -intelssd 1 -  
eventlog
```

`static checkBooleanOption(booleanOption)`

function to set the default value for a boolean option to True and turn it from a string into a boolean value

Args:

booleanOption: string representation of boolean value for boolean flag

Returns:

Boolean value for boolean flag

static checkDebugOption(debugOptions)

function to set the default value for debugOptions and turn it from a string into a boolean value

Args:

debugOptions: string representation of boolean value for debug flag

Returns:

Boolean value for debug flag

static checkDriveName(driveName)

function to set the default value for driveName

Args:

driveName: Name of device interface to get data from

Returns:

String representation of the drive name

static checkDriveNumber(driveNumber)

function to set the default value for driveNumber and turn it from a string into an int

Args:

driveNumber: string representation of int for drive number

Returns:

Int value for drive number

static checkIdentifier(identifier)

function to set the default value for identifier

Args:

identifier: String for the name of the data set that corresponds to the telemetry pull to be executed

Returns:

String for the name of the data set

static checkIterations(iterations)

function to set the default value for iterations and turn it from a string into an int

Args:

iterations: string representation of int value for the number of iterations

Returns:

Int value for number of iterations

static checkOutFile(outFile)

function to set the default value for outFile

Args:

outFile: Output configuration file suffix (including .ini file extension) where the aggregated data will be stored

Returns:

String representation of the output configuration file suffix (including .ini file extension) where the aggregated data will be stored

static checkOutputDir(outputDir)

function to set the default value for outputDir

Args:

outputDir: path to the directory where the binaries will be stored

Returns:

String representation of the path to the output directory

static createDir(dirName)

function for making a directory inside cwd using the specified name in dirName

Args:

dirName: String for path to destination directory

Returns:

static findExecutable(executable='', path=None)

function for finding 'executable' in the directories listed in 'path'.

Args:

executable: the name of the executable file path: A string listing directories separated by 'os.pathsep'; defaults to os.environ['PATH'].

Returns:

the complete filename or None if not found.

static processTextFile(openFile, resultDict, currentName, currentTime)

function that extracts the content of a file into a dictionary to be transferred into the configuration file

Args:

openFile: File descriptor for the open file resultDict: Dictionary containing the objects and fields previously found currentName: String for the current object being processed currentTime: Datetime object containing the time stamp for the current object being processed

Returns:

`src.software.MEP.loadAndProbeDrive.main()`

main function to be called when the script is directly executed from the command line

`src.software.MEP.loadAndProbeDrive.parseInputs()`

`loadAndProbeSystem.py`

This module contains the basic functions for generating a load, preparing a drive and pulling telemetry data from it. It uses IOMeter when running on Windows and FIO when running in Linux. The input file must be a csv file when running on Windows and a bash script when running on Linux. The drive number of the device to be tested must be passed in both, the configuration file and as an argument. The configuration files must have time-based execution for proper telemetry extraction.

Args:

--driveNumber: Integer for the drive number from which to pull telemetry data
--driveName: String for name of device interface to get data from
--inputFile: String for the path to the input file where the workload configuration is stored
--identifier: String for the name of the data set that corresponds to the telemetry pull to be executed
--outputDir: String for the path to the output directory where the binaries will be stored
--parse: Boolean flag to parse the telemetry binaries pulled from the drive
--volumeLabel: String for the label to be used on the disk volume
--volumeAllocUnit: String for the volume allocation unit size
--volumeFS: String for the name of the file system to be used in the disk volume
--volumeLetter: String for the letter to be assigned to the disk volume
--partitionStyle: String for the name of the partition style to be used in the specified disk
--partitionFlag: Boolean flag to indicate if the program should partition the drive using the given parameters
--prepFlag: Boolean flag to indicate if the program should prep the drive before loading it
--debug: Boolean flag to activate verbose printing for debug use

Example:

Default usage (Windows - must be run as administrator):

```
$ python loadAndProbeSystem.py
```

Default usage (Linux):

```
$ sudo python loadAndProbeSystem.py
```

Specific usage (Windows - must be run as administrator):

```
$ python loadAndProbeSystem.py --driveNumber 1 --inputFile
Thermal-4KSW1QD1W.csv --identifier Tv2Hi
--parse True --outputDir binaries --debug True
```

Specific usage (Linux):

```
$ sudo python loadAndProbeSystem.py --driveNumber 0 --inputFile
rand-write.sh --identifier Tv2Hi --parse True
--outputDir binaries --debug True
```

`src.software.TSV.loadAndProbeSystem.API(options=None)`

`src.software.TSV.loadAndProbeSystem.inputsAPI(driveNumber=0, driveName=None, inputFile=None, identifier='Tv2HiTAC', outputDir='.', parse=False, volumeLabel='perftest', volumeAllocUnit='4096', volumeFS='ntfs', volumeLetter='g', partitionStyle='mbr', partitionFlag=True, prepFlag=True, debug=False)`

`class src.software.TSV.loadAndProbeSystem.loadSystem(inputFile, driveNumber, driveName, identifier, outputDir, debug)`

Bases: `object`

`loadDriveWindows(testSchedule)`

function for loading the drive and executing the performance test

Args:

`testSchedule: IometerTestSchedule object containing the input file`

Returns:

```
loadSystemAPI(parse=False, volumeLabel='PERFTEST',  
volumeAllocUnit='4096', volumeFS='ntfs', volumeLetter='G',  
partitionStyle='mbr', partitionFlag=True, prepFlag=True)
```

API to replace standard command line call (the loadSystem class has to instantiated before calling this method). This function also checks that the input file exists and it has the correct suffix

Args:

parse: Boolean flag to parse the telemetry binaries pulled from the drive
volumeLabel: (Windows) String for the label to be used on the disk volume
volumeAllocUnit: (Windows) String for the volume allocation unit size
volumeFS: (Windows) String for the name of the file system to be used in the disk volume
volumeLetter: (Windows) String for the letter to be assigned to the disk volume
partitionStyle: (Windows) String for the name of the partition style to be used in the specified disk
partitionFlag: (Windows) Boolean flag to indicate if the program should partition the drive using the given

parameters

prepFlag: (Linux) Boolean flag to indicate if the program should prep the drive before loading it

Returns:

```
partitionDriveWindows(volumeLabel='PERFTEST',  
volumeAllocUnit='4096', volumeFS='ntfs', volumeLetter='G',  
partitionStyle='mbr')
```

function for partitioning the specified drive using an automatically generated diskpart script for Windows

Args:

volumeLabel: String for the label to be used on the disk volume
volumeAllocUnit: String for the volume allocation unit size
volumeFS: String for the name of the file system to be used in the disk volume
volumeLetter: String for the letter to be

assigned to the disk volume partitionStyle: String for the name of the partition style to be used in the specified disk

Returns:

`prepAndLoadDriveLinux(parse=False, prepFlag=True)`

Driver function for Linux. It calls on FIO to prepare the drives and execute the performance test in an independent subprocess, while the main process collects telemetry via generateTSBinaries

Args:

`parse`: Boolean flag to parse the telemetry binaries pulled from the drive
`prepFlag`: Boolean flag to indicate if the program should prep the drive before loading it

Returns:

`prepAndLoadDriveWindows(parse=False, volumeLabel='PERFTEST', volumeAllocUnit='4096', volumeFS='ntfs', volumeLetter='G', partitionStyle='mbr', partitionFlag=True)`

Driver function for windows. It calls on IOMeter to prepare the drives and execute the performance test in an independent process, while the main process collects telemetry via generateTSBinaries

Args:

`parse`: Boolean flag to parse the telemetry binaries pulled from the drive
`volumeLabel`: String for the label to be used on the disk volume
`volumeAllocUnit`: String for the volume allocation unit size
`volumeFS`: String for the name of the file system to be used in the disk volume
`volumeLetter`: String for the letter to be assigned to the disk volume
`partitionStyle`: String for the name of the partition style to be used in the specified disk
`partitionFlag`: Boolean flag to indicate if the program should partition the drive using the given parameters

Returns:

`prepDriveLinux()`

function for prepping the specified drive using an automatically generated FIO script

Returns:

`prepDriveWindows()`

function for prepping the specified drive for testing. It uses the input file specified during initialization to obtain the configuration settings.

Returns:

testing: IometerTestSchedule object

`src.software.TSV.loadAndProbeSystem.main()`

main function to be called when the script is directly executed from the command line

`src.software.TSV.loadAndProbeSystem.parseInputs()`

class `src.software.TSV.genericObjectGUI.Fields(value)`

Bases: `enum`.`Enum`

An enumeration.

`SECONDARY_VARS = 1`

`TRACKING_VARS = 0`

class `src.software.TSV.genericObjectGUI.FieldsAttributes`

Bases: `object`

`FieldsLabels = {<Fields.TRACKING_VARS: 0>: 'Select tracking variables', <Fields.SECONDARY_VARS: 1>: 'Select optional secondary variables'}`

`FieldsLegendLocation = {<Fields.TRACKING_VARS: 0>: (0, 1.04), <Fields.SECONDARY_VARS: 1>: (1, 1.04)}`

```
FieldsSelectionMode = {<Fields.TRACKING_VARS: 0>: 'multiple',
<Fields.SECONDARY_VARS: 1>: 'multiple'}
```

```
class
src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow(*args
, **kwargs)
Bases: tkinter.Tk
```

Initialization function that initializes the Tk framework

Args:

`*args(list)`: The first parameter containing the variable list of arguments
`**kwargs(dict)`: The second parameter contained named variable arguments.

`after(ms, func=None, *args)`

Call function once after given time.

MS specifies the time in milliseconds. FUNC gives the function which shall be called. Additional parameters are given as parameters to the function call. Return identifier to cancel scheduling with `after_cancel`.

`after_cancel(id)`

Cancel scheduling of function identified with ID.

Identifier returned by `after` or `after_idle` must be given as first parameter.

`after_idle(func, *args)`

Call FUNC once if the Tcl main loop has no event to process.

Return an identifier to cancel the scheduling with `after_cancel`.

`anchor(anchor=None)`

The anchor value controls how to place the grid within the master when no row/column has any weight.

The default anchor is nw.

aspect(*minNumer=None*, *minDenom=None*, *maxNumer=None*,
maxDenom=None)

Instruct the window manager to set the aspect ratio (width/height) of this widget to be between MINNUMER/MINDENOM and MAXNUMBER/MAXDENOM. Return a tuple of the actual values if no argument is given.

attributes(**args*)

This subcommand returns or sets platform specific attributes

The first form returns a list of the platform specific flags and their values. The second form returns the value for the specific option. The third form sets one or more of the values. The values are as follows:

On Windows, -disabled gets or sets whether the window is in a disabled state. -toolwindow gets or sets the style of the window to toolwindow (as defined in the MSDN). -topmost gets or sets whether this is a topmost window (displays above all other windows).

On Macintosh, XXXXX

On Unix, there are currently no special attribute values.

bbox(*column=None*, *row=None*, *col2=None*, *row2=None*)

Return a tuple of integer coordinates for the bounding box of this widget controlled by the geometry manager grid.

If COLUMN, ROW is given the bounding box applies from the cell with row and column 0 to the specified cell. If COL2 and ROW2 are given the bounding box starts at that cell.

The returned integers specify the offset of the upper left corner in the master widget and the width and height.

`bell(displayof=0)`

Ring a display's bell.

`bind(sequence=None, func=None, add=None)`

Bind to this widget at event SEQUENCE a call to function FUNC.

SEQUENCE is a string of concatenated event patterns. An event pattern is of the form <MODIFIER-MODIFIER-TYPE-DETAIL> where MODIFIER is one of Control, Mod2, M2, Shift, Mod3, M3, Lock, Mod4, M4, Button1, B1, Mod5, M5 Button2, B2, Meta, M, Button3, B3, Alt, Button4, B4, Double, Button5, B5 Triple, Mod1, M1. TYPE is one of Activate, Enter, Map, ButtonPress, Button, Expose, Motion, ButtonRelease FocusIn, MouseWheel, Circulate, FocusOut, Property, Colormap, Gravity Reparent, Configure, KeyPress, Key, Unmap, Deactivate, KeyRelease Visibility, Destroy, Leave and DETAIL is the button number for ButtonPress, ButtonRelease and DETAIL is the Keysym for KeyPress and KeyRelease. Examples are <Control-Button-1> for pressing Control and mouse button 1 or <Alt-A> for pressing A and the Alt key (KeyPress can be omitted). An event pattern can also be a virtual event of the form <<AString>> where AString can be arbitrary. This event can be generated by event_generate. If events are concatenated they must appear shortly after each other.

FUNC will be called if the event sequence occurs with an instance of Event as argument. If the return value of FUNC is "break" no further bound function is invoked.

An additional boolean parameter ADD specifies whether FUNC will be called additionally to the other bound function or whether it will replace the previous function.

Bind will return an identifier to allow deletion of the bound function with unbind without memory leak.

If FUNC or SEQUENCE is omitted the bound function or list of bound events are returned.

`bind_all(sequence=None, func=None, add=None)`

Bind to all widgets at an event SEQUENCE a call to function FUNC. An additional boolean parameter ADD specifies whether FUNC will be called additionally to the other bound function or whether it will replace the previous function. See bind for the return value.

`bind_class(className, sequence=None, func=None, add=None)`

Bind to widgets with bindtag CLASSNAME at event SEQUENCE a call of function FUNC. An additional boolean parameter ADD specifies whether FUNC will be called additionally to the other bound function or whether it will replace the previous function. See bind for the return value.

`bindtags(tagList=None)`

Set or get the list of bindtags for this widget.

With no argument return the list of all bindtags associated with this widget. With a list of strings as argument the bindtags are set to this list. The bindtags determine in which order events are processed (see bind).

`cget(key)`

Return the resource value for a KEY given as string.

`client(name=None)`

Store NAME in WM_CLIENT_MACHINE property of this widget.
Return current value.

`clipboard_append(string, **kw)`

Append STRING to the Tk clipboard.

A widget specified at the optional displayof keyword argument specifies the target display. The clipboard can be retrieved with selection_get.

`clipboard_clear(**kw)`

Clear the data in the Tk clipboard.

A widget specified for the optional displayof keyword argument specifies the target display.

`clipboard_get(**kw)`

Retrieve data from the clipboard on window's display.

The window keyword defaults to the root window of the Tkinter application.

The type keyword specifies the form in which the data is to be returned and should be an atom name such as STRING or FILE_NAME. Type defaults to STRING, except on X11, where the default is to try UTF8_STRING and fall back to STRING.

This command is equivalent to:

`selection_get(CLIPBOARD)`

`colormapwindows(*wlist)`

Store list of window names (WLIST) into WM_COLORMAPWINDOWS property of this widget. This list contains windows whose colormaps differ from their parents. Return current list of widgets if WLIST is empty.

`columnconfigure(index, cnf={}, **kw)`

Configure column INDEX of a grid.

Valid resources are minsize (minimum size of the column), weight (how much does additional space propagate to this column) and pad (how much space to let additionally).

`command(value=None)`

Store VALUE in WM_COMMAND property. It is the command which shall be used to invoke the application. Return current

command if VALUE is None.

`config(cnf=None, **kw)`

Configure resources of a widget.

The values for resources are specified as keyword arguments. To get an overview about the allowed keyword arguments call the method `keys`.

`configure(cnf=None, **kw)`

Configure resources of a widget.

The values for resources are specified as keyword arguments. To get an overview about the allowed keyword arguments call the method `keys`.

`deiconify()`

Deiconify this widget. If it was never mapped it will not be mapped. On Windows it will raise this widget and give it the focus.

`deletecommand(name)`

Internal function.

Delete the Tcl command provided in NAME.

`destroy()`

Destroy this and all descendants widgets. This will end the application of this Tcl interpreter.

`event_add(virtual, *sequences)`

Bind a virtual event VIRTUAL (of the form <<Name>>) to an event SEQUENCE such that the virtual event is triggered whenever SEQUENCE occurs.

`event_delete(virtual, *sequences)`

Unbind a virtual event VIRTUAL from SEQUENCE.

`event_generate(sequence, **kw)`

Generate an event SEQUENCE. Additional keyword arguments specify parameter of the event (e.g. x, y, rootx, rooty).

`event_info(virtual=None)`

Return a list of all virtual events or the information about the SEQUENCE bound to the virtual event VIRTUAL.

`focus()`

Direct input focus to this widget.

If the application currently does not have the focus this widget will get the focus if the application gets the focus through the window manager.

`focus_displayof()`

Return the widget which has currently the focus on the display where this widget is located.

Return None if the application does not have the focus.

`focus_force()`

Direct input focus to this widget even if the application does not have the focus. Use with caution!

`focus_get()`

Return the widget which has currently the focus in the application.

Use `focus_displayof` to allow working with several displays. Return None if application does not have the focus.

`focus_lastfor()`

Return the widget which would have the focus if top level for this widget gets the focus from the window manager.

`focus_set()`

Direct input focus to this widget.

If the application currently does not have the focus this widget will get the focus if the application gets the focus through the window manager.

`focusmodel(model=None)`

Set focus model to MODEL. "active" means that this widget will claim the focus itself, "passive" means that the window manager shall give the focus. Return current focus model if MODEL is None.

`forget(window)`

The window will be unmapped from the screen and will no longer be managed by wm. toplevel windows will be treated like frame windows once they are no longer managed by wm, however, the menu option configuration will be remembered and the menus will return once the widget is managed again.

`frame()`

Return identifier for decorative frame of this widget if present.

`geometry(newGeometry=None)`

Set geometry to NEWGEOMETRY of the form =widthxheight+x+y.
Return current value if None is given.

`getboolean(s)`

Return a boolean value for Tcl boolean values true and false given as parameter.

`getdouble(s)`

`getInt(s)`

`getvar(name='PY_VAR')`

Return value of Tcl variable NAME.

`grab_current()`

Return widget which has currently the grab in this application or None.

`grab_release()`

Release grab for this widget if currently set.

`grab_set()`

Set grab for this widget.

A grab directs all events to this and descendant widgets in the application.

`grab_set_global()`

Set global grab for this widget.

A global grab directs all events to this and descendant widgets on the display. Use with caution - other applications do not get events anymore.

`grab_status()`

Return None, "local" or "global" if this widget has no, a local or a global grab.

`grid(baseWidth=None, baseHeight=None, widthInc=None, heightInc=None)`

Instruct the window manager that this widget shall only be resized on grid boundaries. WIDTHINC and HEIGHTINC are the width and height of a grid unit in pixels. BASEWIDTH and BASEHEIGHT are the number of grid units requested in Tk_GeometryRequest.

`grid_anchor(anchor=None)`

The anchor value controls how to place the grid within the master when no row/column has any weight.

The default anchor is nw.

`grid_bbox(column=None, row=None, col2=None, row2=None)`

Return a tuple of integer coordinates for the bounding box of this widget controlled by the geometry manager grid.

If COLUMN, ROW is given the bounding box applies from the cell with row and column 0 to the specified cell. If COL2 and ROW2 are given the bounding box starts at that cell.

The returned integers specify the offset of the upper left corner in the master widget and the width and height.

`grid_columnconfigure(index, cnf={}, **kw)`

Configure column INDEX of a grid.

Valid resources are minsize (minimum size of the column), weight (how much does additional space propagate to this column) and pad (how much space to let additionally).

`grid_location(x, y)`

Return a tuple of column and row which identify the cell at which the pixel at position X and Y inside the master widget is located.

`grid_propagate(flag=['noarg_'])`

Set or get the status for propagation of geometry information.

A boolean argument specifies whether the geometry information of the slaves will determine the size of this widget. If no argument is given, the current setting will be returned.

`grid_rowconfigure(index, cnf={}, **kw)`

Configure row INDEX of a grid.

Valid resources are minsize (minimum size of the row), weight (how much does additional space propagate to this row) and pad (how much space to let additionally).

`grid_size()`

Return a tuple of the number of column and rows in the grid.

`grid_slaves(row=None, column=None)`

Return a list of all slaves of this widget in its packing order.

`group(pathName=None)`

Set the group leader widgets for related widgets to PATHNAME.

Return the group leader of this widget if None is given.

`iconbitmap(bitmap=None, default=None)`

Set bitmap for the iconified widget to BITMAP. Return the bitmap if None is given.

Under Windows, the DEFAULT parameter can be used to set the icon for the widget and any descendants that don't have an icon set explicitly. DEFAULT can be the relative path to a .ico file (example: root.iconbitmap(default='myicon.ico')). See Tk documentation for more information.

`iconify()`

Display widget as icon.

`iconmask(bitmap=None)`

Set mask for the icon bitmap of this widget. Return the mask if None is given.

`iconname(newName=None)`

Set the name of the icon for this widget. Return the name if None is given.

`iconphoto(default=False, *args)`

Sets the titlebar icon for this window based on the named photo images passed through args. If default is True, this is applied to all future created toplevels as well.

The data in the images is taken as a snapshot at the time of invocation. If the images are later changed, this is not reflected to the titlebar icons. Multiple images are accepted to allow different images sizes to be provided. The window manager may scale provided icons to an appropriate size.

On Windows, the images are packed into a Windows icon structure. This will override an icon specified to `wm_iconbitmap`, and vice versa.

On X, the images are arranged into the `_NET_WM_ICON` X property, which most modern window managers support. An icon specified by `wm_iconbitmap` may exist simultaneously.

On Macintosh, this currently does nothing.

`iconposition(x=None, y=None)`

Set the position of the icon of this widget to X and Y. Return a tuple of the current values of X and Y if None is given.

`iconwindow(pathName=None)`

Set widget PATHNAME to be displayed instead of icon. Return the current value if None is given.

`image_names()`

Return a list of all existing image names.

`image_types()`

Return a list of all available image types (e.g. photo bitmap).

`init_frame()`

`keys()`

Return a list of all resource names of this widget.

`lift(aboveThis=None)`

Raise this widget in the stacking order.

`loadtk()`

`lower(belowThis=None)`

Lower this widget in the stacking order.

`mainloop(n=0)`

Call the mainloop of Tk.

`manage(widget)`

The widget specified will become a stand alone top-level window. The window will be decorated with the window managers title bar, etc.

`maxsize(width=None, height=None)`

Set max WIDTH and HEIGHT for this widget. If the window is gridded the values are given in grid units. Return the current values if None is given.

`minsize(width=None, height=None)`

Set min WIDTH and HEIGHT for this widget. If the window is gridded the values are given in grid units. Return the current values if None is given.

`nametowidget(name)`

Return the Tkinter instance of a widget identified by its Tcl name NAME.

`option_add(pattern, value, priority=None)`

Set a VALUE (second parameter) for an option PATTERN (first parameter).

An optional third parameter gives the numeric priority (defaults to 80).

`option_clear()`

Clear the option database.

It will be reloaded if option_add is called.

option_get(*name, className*)

Return the value for an option NAME for this widget with CLASSNAME.

Values with higher priority override lower values.

option_readfile(*fileName, priority=None*)

Read file FILENAME into the option database.

An optional second parameter gives the numeric priority.

overrideredirect(*boolean=None*)

Instruct the window manager to ignore this widget if BOOLEAN is given with 1. Return the current value if None is given.

pack_propagate(*flag=['_noarg_']*)

Set or get the status for propagation of geometry information.

A boolean argument specifies whether the geometry information of the slaves will determine the size of this widget. If no argument is given the current setting will be returned.

pack_slaves()

Return a list of all slaves of this widget in its packing order.

place_slaves()

Return a list of all slaves of this widget in its packing order.

positionfrom(*who=None*)

Instruct the window manager that the position of this widget shall be defined by the user if WHO is "user", and by its own policy if WHO is "program".

propagate(*flag=['_noarg_']*)

Set or get the status for propagation of geometry information.

A boolean argument specifies whether the geometry information of the slaves will determine the size of this widget. If no argument is given the current setting will be returned.

`protocol(name=None, func=None)`

Bind function FUNC to command NAME for this widget. Return the function bound to NAME if None is given. NAME could be e.g. "WM_SAVE_YOURSELF" or "WM_DELETE_WINDOW".

`quit()`

Quit the Tcl interpreter. All widgets will be destroyed.

`readprofile(baseName, className)`

Internal function. It reads BASENAME.tcl and CLASSNAME.tcl into the Tcl Interpreter and calls exec on the contents of BASENAME.py and CLASSNAME.py if such a file exists in the home directory.

`register(func, subst=None, needcleanup=1)`

Return a newly created Tcl function. If this function is called, the Python function FUNC will be executed. An optional function SUBST can be given which will be executed before FUNC.

`report_callback_exception(exc, val, tb)`

Report callback exception on sys.stderr.

Applications may want to override this internal function, and should when sys.stderr is None.

`resizable(width=None, height=None)`

Instruct the window manager whether this width can be resized in WIDTH or HEIGHT. Both values are boolean values.

`rowconfigure(index, cnf={}, **kw)`

Configure row INDEX of a grid.

Valid resources are minsize (minimum size of the row), weight (how much does additional space propagate to this row) and pad (how much space to let additionally).

`selection_clear(**kw)`

Clear the current X selection.

`selection_get(**kw)`

Return the contents of the current X selection.

A keyword parameter selection specifies the name of the selection and defaults to PRIMARY. A keyword parameter displayof specifies a widget on the display to use. A keyword parameter type specifies the form of data to be fetched, defaulting to STRING except on X11, where UTF8_STRING is tried before STRING.

`selection_handle(command, **kw)`

Specify a function COMMAND to call if the X selection owned by this widget is queried by another application.

This function must return the contents of the selection. The function will be called with the arguments OFFSET and LENGTH which allows the chunking of very long selections. The following keyword parameters can be provided: selection - name of the selection (default PRIMARY), type - type of the selection (e.g. STRING, FILE_NAME).

`selection_own(**kw)`

Become owner of X selection.

A keyword parameter selection specifies the name of the selection (default PRIMARY).

`selection_own_get(**kw)`

Return owner of X selection.

The following keyword parameter can be provided: selection - name of the selection (default PRIMARY), type - type of the selection (e.g. STRING, FILE_NAME).

`send(interp, cmd, *args)`

Send Tcl command CMD to different interpreter INTERP to be executed.

`setvar(name='PY_VAR', value='I')`

Set Tcl variable NAME to VALUE.

`show_frame()`

`size()`

Return a tuple of the number of column and rows in the grid.

`sizefrom(who=None)`

Instruct the window manager that the size of this widget shall be defined by the user if WHO is "user", and by its own policy if WHO is "program".

`slaves()`

Return a list of all slaves of this widget in its packing order.

`state(newstate=None)`

Query or set the state of this widget as one of normal, icon, iconic (see `wm_iconwindow`), withdrawn, or zoomed (Windows only).

`title(string=None)`

Set the title of this widget.

`tk_bisque()`

Change the color scheme to light brown as used in Tk 3.6 and before.

`tk_focusFollowsMouse()`

The widget under mouse will get automatically focus. Can not be disabled easily.

`tk_focusNext()`

Return the next widget in the focus order which follows widget which has currently the focus.

The focus order first goes to the next child, then to the children of the child recursively and then to the next sibling which is higher in the stacking order. A widget is omitted if it has the `takefocus` resource set to 0.

`tk_focusPrev()`

Return previous widget in the focus order. See `tk_focusNext` for details.

`tk_setPalette(*args, **kw)`

Set a new color scheme for all widget elements.

A single color as argument will cause that all colors of Tk widget elements are derived from this. Alternatively several keyword parameters and its associated colors can be given. The following keywords are valid: `activeBackground`, `foreground`, `selectColor`, `activeForeground`, `highlightBackground`, `selectBackground`, `background`, `highlightColor`, `selectForeground`, `disabledForeground`, `insertBackground`, `troughColor`.

`tk_strictMotif(boolean=None)`

Set Tcl internal variable, whether the look and feel should adhere to Motif.

A parameter of 1 means adhere to Motif (e.g. no color change if mouse passes over slider). Returns the set value.

`tkraise(aboveThis=None)`

Raise this widget in the stacking order.

`transient(master=None)`

Instruct the window manager that this widget is transient with regard to widget MASTER.

`unbind(sequence, funcid=None)`

Bind for this widget for event SEQUENCE the function identified with FUNCID.

`unbind_all(sequence)`

Bind for all widgets for event SEQUENCE all functions.

`unbind_class(className, sequence)`

Bind for all widgets with bindtag CLASSNAME for event SEQUENCE all functions.

`update()`

Enter event loop until all pending events have been processed by Tcl.

`update_idletasks()`

Enter event loop until all idle callbacks have been called. This will update the display of windows but not process events caused by the user.

`wait_variable(name='PY_VAR')`

Wait until the variable is modified.

A parameter of type IntVar, StringVar, DoubleVar or BooleanVar must be given.

`wait_visibility(window=None)`

Wait until the visibility of a WIDGET changes (e.g. it appears).

If no parameter is given self is used.

`wait_window(window=None)`

Wait until a WIDGET is destroyed.

If no parameter is given self is used.

`waitvar(name='PY_VAR')`

Wait until the variable is modified.

A parameter of type IntVar, StringVar, DoubleVar or BooleanVar must be given.

`winfo_atom(name, displayof=0)`

Return integer which represents atom NAME.

`winfo_atomname(id, displayof=0)`

Return name of atom with identifier ID.

`winfo_cells()`

Return number of cells in the colormap for this widget.

`winfo_children()`

Return a list of all widgets which are children of this widget.

`winfo_class()`

Return window class name of this widget.

`winfo_colormapfull()`

Return True if at the last color request the colormap was full.

`winfo_containing(rootX, rootY, displayof=0)`

Return the widget which is at the root coordinates ROOTX, ROOTY.

`winfo_depth()`

Return the number of bits per pixel.

`winfo_exists()`

Return true if this widget exists.

winfo_fpixels(*number*)

Return the number of pixels for the given distance NUMBER (e.g. "3c") as float.

winfo_geometry()

Return geometry string for this widget in the form "widthxheight+X+Y".

winfo_height()

Return height of this widget.

winfo_id()

Return identifier ID for this widget.

winfo_interps(*displayof*=0)

Return the name of all Tcl interpreters for this display.

winfo_ismapped()

Return true if this widget is mapped.

winfo_manager()

Return the window manager name for this widget.

winfo_name()

Return the name of this widget.

winfo_parent()

Return the name of the parent of this widget.

winfo_pathname(*id, displayof*=0)

Return the pathname of the widget given by ID.

winfo_pixels(*number*)

Rounded integer value of winfo_fpixels.

winfo_pointerx()

Return the x coordinate of the pointer on the root window.

winfo_pointerxy()

Return a tuple of x and y coordinates of the pointer on the root window.

winfo_pointery()

Return the y coordinate of the pointer on the root window.

winfo_reqheight()

Return requested height of this widget.

winfo_reqwidth()

Return requested width of this widget.

winfo_rgb(*color*)

Return a tuple of integer RGB values in range(65536) for color in this widget.

winfo_rootx()

Return x coordinate of upper left corner of this widget on the root window.

winfo_rooty()

Return y coordinate of upper left corner of this widget on the root window.

winfo_screen()

Return the screen name of this widget.

winfo_screencells()

Return the number of the cells in the colormap of the screen of this widget.

winfo_screendepth()

Return the number of bits per pixel of the root window of the screen of this widget.

winfo_screenheight()

Return the number of pixels of the height of the screen of this widget in pixel.

winfo_screenmmheight()

Return the number of pixels of the height of the screen of this widget in mm.

winfo_screenmmwidth()

Return the number of pixels of the width of the screen of this widget in mm.

winfo_screenvisual()

Return one of the strings directcolor, grayscale, pseudocolor, staticcolor, staticgray, or truecolor for the default colormodel of this screen.

winfo_screenwidth()

Return the number of pixels of the width of the screen of this widget in pixel.

winfo_server()

Return information of the X-Server of the screen of this widget in the form "XmajorRminor vendor vendorVersion".

winfo_toplevel()

Return the toplevel widget of this widget.

winfo_viewable()

Return true if the widget and all its higher ancestors are mapped.

winfo_visual()

Return one of the strings directcolor, grayscale, pseudocolor, staticcolor, staticgray, or truecolor for the colormodel of this widget.

winfo_visualid()

Return the X identifier for the visual for this widget.

winfo_visualsavailable(*includeids=False*)

Return a list of all visuals available for the screen of this widget.

Each item in the list consists of a visual name (see winfo_visual), a depth and if includeids is true is given also the X identifier.

winfo_vrootheight()

Return the height of the virtual root window associated with this widget in pixels. If there is no virtual root window return the height of the screen.

winfo_vrootwidth()

Return the width of the virtual root window associated with this widget in pixel. If there is no virtual root window return the width of the screen.

winfo_vrootx()

Return the x offset of the virtual root relative to the root window of the screen of this widget.

winfo_vrooty()

Return the y offset of the virtual root relative to the root window of the screen of this widget.

winfo_width()

Return the width of this widget.

`winfo_x()`

Return the x coordinate of the upper left corner of this widget in the parent.

`winfo_y()`

Return the y coordinate of the upper left corner of this widget in the parent.

`withdraw()`

Withdraw this widget from the screen such that it is unmapped and forgotten by the window manager. Re-draw it with `wm_deiconify`.

`wm_aspect(minNumer=None, minDenom=None, maxNumer=None, maxDenom=None)`

Instruct the window manager to set the aspect ratio (width/height) of this widget to be between MINNUMER/MINDENOM and MAXNUMER/MAXDENOM. Return a tuple of the actual values if no argument is given.

`wm_attributes(*args)`

This subcommand returns or sets platform specific attributes

The first form returns a list of the platform specific flags and their values. The second form returns the value for the specific option. The third form sets one or more of the values. The values are as follows:

On Windows, `-disabled` gets or sets whether the window is in a disabled state. `-toolwindow` gets or sets the style of the window to toolwindow (as defined in the MSDN). `-topmost` gets or sets whether this is a topmost window (displays above all other windows).

On Macintosh, XXXXX

On Unix, there are currently no special attribute values.

`wm_client(name=None)`

Store NAME in WM_CLIENT_MACHINE property of this widget.
Return current value.

`wm_colormapwindows(*wlist)`

Store list of window names (WLIST) into
WM_COLORMAPWINDOWS property of this widget. This list
contains windows whose colormaps differ from their parents. Return
current list of widgets if WLIST is empty.

`wm_command(value=None)`

Store VALUE in WM_COMMAND property. It is the command
which shall be used to invoke the application. Return current
command if VALUE is None.

`wm_deiconify()`

Deiconify this widget. If it was never mapped it will not be mapped.
On Windows it will raise this widget and give it the focus.

`wm_focusmodel(model=None)`

Set focus model to MODEL. "active" means that this widget will
claim the focus itself, "passive" means that the window manager
shall give the focus. Return current focus model if MODEL is None.

`wm_forget(window)`

The window will be unmapped from the screen and will no longer
be managed by wm. toplevel windows will be treated like frame
windows once they are no longer managed by wm, however, the
menu option configuration will be remembered and the menus will
return once the widget is managed again.

`wm_frame()`

Return identifier for decorative frame of this widget if present.

`wm_geometry(newGeometry=None)`

Set geometry to NEWGEOMETRY of the form =widthxheight+x+y.
Return current value if None is given.

`wm_grid(baseWidth=None, baseHeight=None, widthInc=None,
heightInc=None)`

Instruct the window manager that this widget shall only be resized on grid boundaries. WIDTHINC and HEIGHTINC are the width and height of a grid unit in pixels. BASEWIDTH and BASEHEIGHT are the number of grid units requested in Tk_GeometryRequest.

`wm_group(pathName=None)`

Set the group leader widgets for related widgets to PATHNAME.
Return the group leader of this widget if None is given.

`wm_iconbitmap(bitmap=None, default=None)`

Set bitmap for the iconified widget to BITMAP. Return the bitmap if None is given.

Under Windows, the DEFAULT parameter can be used to set the icon for the widget and any descendants that don't have an icon set explicitly. DEFAULT can be the relative path to a .ico file (example: root.iconbitmap(default='myicon.ico')). See Tk documentation for more information.

`wm_iconify()`

Display widget as icon.

`wm_iconmask(bitmap=None)`

Set mask for the icon bitmap of this widget. Return the mask if None is given.

`wm_iconname(newName=None)`

Set the name of the icon for this widget. Return the name if None is given.

`wm_iconphoto(default=False, *args)`

Sets the titlebar icon for this window based on the named photo images passed through args. If default is True, this is applied to all future created toplevels as well.

The data in the images is taken as a snapshot at the time of invocation. If the images are later changed, this is not reflected to the titlebar icons. Multiple images are accepted to allow different images sizes to be provided. The window manager may scale provided icons to an appropriate size.

On Windows, the images are packed into a Windows icon structure. This will override an icon specified to `wm_iconbitmap`, and vice versa.

On X, the images are arranged into the `_NET_WM_ICON` X property, which most modern window managers support. An icon specified by `wm_iconbitmap` may exist simultaneously.

On Macintosh, this currently does nothing.

`wm_iconposition(x=None, y=None)`

Set the position of the icon of this widget to X and Y. Return a tuple of the current values of X and Y if None is given.

`wm_iconwindow(pathName=None)`

Set widget PATHNAME to be displayed instead of icon. Return the current value if None is given.

`wm_manage(widget)`

The widget specified will become a stand alone top-level window. The window will be decorated with the window managers title bar, etc.

`wm_maxsize(width=None, height=None)`

Set max WIDTH and HEIGHT for this widget. If the window is gridded the values are given in grid units. Return the current values

if None is given.

`wm_minsize(width=None, height=None)`

Set min WIDTH and HEIGHT for this widget. If the window is gridded the values are given in grid units. Return the current values if None is given.

`wm_overrideredirect(boolean=None)`

Instruct the window manager to ignore this widget if BOOLEAN is given with 1. Return the current value if None is given.

`wm_positionfrom(who=None)`

Instruct the window manager that the position of this widget shall be defined by the user if WHO is "user", and by its own policy if WHO is "program".

`wm_protocol(name=None, func=None)`

Bind function FUNC to command NAME for this widget. Return the function bound to NAME if None is given. NAME could be e.g. "WM_SAVE_YOURSELF" or "WM_DELETE_WINDOW".

`wm_resizable(width=None, height=None)`

Instruct the window manager whether this width can be resized in WIDTH or HEIGHT. Both values are boolean values.

`wm_sizefrom(who=None)`

Instruct the window manager that the size of this widget shall be defined by the user if WHO is "user", and by its own policy if WHO is "program".

`wm_state(newstate=None)`

Query or set the state of this widget as one of normal, icon, iconic (see `wm_iconwindow`), withdrawn, or zoomed (Windows only).

`wm_title(string=None)`

Set the title of this widget.

`wm_transient(master=None)`

Instruct the window manager that this widget is transient with regard to widget MASTER.

`wm_withdraw()`

Withdraw this widget from the screen such that it is unmapped and forgotten by the window manager. Re-draw it with `wm_deiconify`.

`class src.software.TSV.genericObjectGUI.ObjectConfig(configFilePath)`

Bases: `object`

`objectFilePath = None`

`objectIDs = []`

`readConfigContent(debug=False)`

`trackingVars = []`

`class src.software.TSV.genericObjectGUI.StartPage(parent, controller)`

Bases: `tkinter.Frame`

`after(ms, func=None, *args)`

Call function once after given time.

MS specifies the time in milliseconds. FUNC gives the function which shall be called. Additional parameters are given as parameters to the function call. Return identifier to cancel scheduling with `after_cancel`.

`after_cancel(id)`

Cancel scheduling of function identified with ID.

Identifier returned by `after` or `after_idle` must be given as first parameter.

`after_idle(func, *args)`

Call FUNC once if the Tcl main loop has no event to process.

Return an identifier to cancel the scheduling with after_cancel.

`anchor(anchor=None)`

The anchor value controls how to place the grid within the master when no row/column has any weight.

The default anchor is nw.

`bbox(column=None, row=None, col2=None, row2=None)`

Return a tuple of integer coordinates for the bounding box of this widget controlled by the geometry manager grid.

If COLUMN, ROW is given the bounding box applies from the cell with row and column 0 to the specified cell. If COL2 and ROW2 are given the bounding box starts at that cell.

The returned integers specify the offset of the upper left corner in the master widget and the width and height.

`bell(displayof=0)`

Ring a display's bell.

`bind(sequence=None, func=None, add=None)`

Bind to this widget at event SEQUENCE a call to function FUNC.

SEQUENCE is a string of concatenated event patterns. An event pattern is of the form <MODIFIER-MODIFIER-TYPE-DETAIL> where MODIFIER is one of Control, Mod2, M2, Shift, Mod3, M3, Lock, Mod4, M4, Button1, B1, Mod5, M5 Button2, B2, Meta, M, Button3, B3, Alt, Button4, B4, Double, Button5, B5 Triple, Mod1, M1. TYPE is one of Activate, Enter, Map, ButtonPress, Button, Expose, Motion, ButtonRelease FocusIn, MouseWheel, Circulate, FocusOut, Property, Colormap, Gravity Reparent, Configure, KeyPress, Key, Unmap, Deactivate, KeyRelease Visibility, Destroy, Leave and DETAIL is the button number for ButtonPress,

ButtonRelease and DETAIL is the Keysym for KeyPress and KeyRelease. Examples are <Control-Button-1> for pressing Control and mouse button 1 or <Alt-A> for pressing A and the Alt key (KeyPress can be omitted). An event pattern can also be a virtual event of the form <<AString>> where AString can be arbitrary. This event can be generated by event_generate. If events are concatenated they must appear shortly after each other.

FUNC will be called if the event sequence occurs with an instance of Event as argument. If the return value of FUNC is "break" no further bound function is invoked.

An additional boolean parameter ADD specifies whether FUNC will be called additionally to the other bound function or whether it will replace the previous function.

Bind will return an identifier to allow deletion of the bound function with unbind without memory leak.

If FUNC or SEQUENCE is omitted the bound function or list of bound events are returned.

bind_all(*sequence=None, func=None, add=None*)

Bind to all widgets at an event SEQUENCE a call to function FUNC. An additional boolean parameter ADD specifies whether FUNC will be called additionally to the other bound function or whether it will replace the previous function. See bind for the return value.

bind_class(*className, sequence=None, func=None, add=None*)

Bind to widgets with bindtag CLASSNAME at event SEQUENCE a call of function FUNC. An additional boolean parameter ADD specifies whether FUNC will be called additionally to the other bound function or whether it will replace the previous function. See bind for the return value.

bindtags(*tagList=None*)

Set or get the list of bindtags for this widget.

With no argument return the list of all bindtags associated with this widget. With a list of strings as argument the bindtags are set to this list. The bindtags determine in which order events are processed (see bind).

`cget(key)`

Return the resource value for a KEY given as string.

`clipboard_append(string, **kw)`

Append STRING to the Tk clipboard.

A widget specified at the optional displayof keyword argument specifies the target display. The clipboard can be retrieved with `selection_get`.

`clipboard_clear(**kw)`

Clear the data in the Tk clipboard.

A widget specified for the optional displayof keyword argument specifies the target display.

`clipboard_get(**kw)`

Retrieve data from the clipboard on window's display.

The window keyword defaults to the root window of the Tkinter application.

The type keyword specifies the form in which the data is to be returned and should be an atom name such as STRING or FILE_NAME. Type defaults to STRING, except on X11, where the default is to try UTF8_STRING and fall back to STRING.

This command is equivalent to:

`selection_get(CLIPBOARD)`

`columnconfigure(index, cnf={}, **kw)`

Configure column INDEX of a grid.

Valid resources are minsize (minimum size of the column), weight (how much does additional space propagate to this column) and pad (how much space to let additionally).

`config(cnf=None, **kw)`

Configure resources of a widget.

The values for resources are specified as keyword arguments. To get an overview about the allowed keyword arguments call the method keys.

`configure(cnf=None, **kw)`

Configure resources of a widget.

The values for resources are specified as keyword arguments. To get an overview about the allowed keyword arguments call the method keys.

`create_new_graph_window()`

`deletecommand(name)`

Internal function.

Delete the Tcl command provided in NAME.

`destroy()`

Destroy this and all descendants widgets.

`display_dir()`

`event_add(virtual, *sequences)`

Bind a virtual event VIRTUAL (of the form <<Name>>) to an event SEQUENCE such that the virtual event is triggered whenever SEQUENCE occurs.

`event_delete(virtual, *sequences)`

Unbind a virtual event VIRTUAL from SEQUENCE.

`event_generate(sequence, **kw)`

Generate an event SEQUENCE. Additional keyword arguments specify parameter of the event (e.g. x, y, rootx, rooty).

`event_info(virtual=None)`

Return a list of all virtual events or the information about the SEQUENCE bound to the virtual event VIRTUAL.

`focus()`

Direct input focus to this widget.

If the application currently does not have the focus this widget will get the focus if the application gets the focus through the window manager.

`focus_displayof()`

Return the widget which has currently the focus on the display where this widget is located.

Return None if the application does not have the focus.

`focus_force()`

Direct input focus to this widget even if the application does not have the focus. Use with caution!

`focus_get()`

Return the widget which has currently the focus in the application.

Use `focus_displayof` to allow working with several displays. Return None if application does not have the focus.

`focus_lastfor()`

Return the widget which would have the focus if top level for this widget gets the focus from the window manager.

`focus_set()`

Direct input focus to this widget.

If the application currently does not have the focus this widget will get the focus if the application gets the focus through the window manager.

`forget()`

Unmap this widget and do not use it for the packing order.

`get_listbox_content(listbox)`

`getboolean(s)`

Return a boolean value for Tcl boolean values true and false given as parameter.

`getdouble(s)`

`getint(s)`

`getvar(name='PY_VAR')`

Return value of Tcl variable NAME.

`grab_current()`

Return widget which has currently the grab in this application or None.

`grab_release()`

Release grab for this widget if currently set.

`grab_set()`

Set grab for this widget.

A grab directs all events to this and descendant widgets in the application.

`grab_set_global()`

Set global grab for this widget.

A global grab directs all events to this and descendant widgets on the display. Use with caution - other applications do not get events anymore.

`grab_status()`

Return None, "local" or "global" if this widget has no, a local or a global grab.

`grid(cnf={}, **kw)`

Position a widget in the parent widget in a grid. Use as options:
column=number - use cell identified with given column (starting with 0)
columnspan=number - this widget will span several columns
in=master - use master to contain this widget in_=master - see 'in'
option description ipadx=amount - add internal padding in x direction
ipady=amount - add internal padding in y direction
padx=amount - add padding in x direction
pady=amount - add padding in y direction
row=number - use cell identified with given row (starting with 0)
rowspan=number - this widget will span several rows
sticky=NSEW - if cell is larger on which sides will this

widget stick to the cell boundary

`grid_anchor(anchor=None)`

The anchor value controls how to place the grid within the master when no row/column has any weight.

The default anchor is nw.

`grid_bbox(column=None, row=None, col2=None, row2=None)`

Return a tuple of integer coordinates for the bounding box of this widget controlled by the geometry manager grid.

If COLUMN, ROW is given the bounding box applies from the cell with row and column 0 to the specified cell. If COL2 and ROW2 are given the bounding box starts at that cell.

The returned integers specify the offset of the upper left corner in the master widget and the width and height.

`grid_columnconfigure(index, cnf={}, **kw)`

Configure column INDEX of a grid.

Valid resources are minsize (minimum size of the column), weight (how much does additional space propagate to this column) and pad (how much space to let additionally).

`grid_configure(cnf={}, **kw)`

Position a widget in the parent widget in a grid. Use as options:
column=number - use cell identified with given column (starting with 0)
columnspan=number - this widget will span several columns
in=master - use master to contain this widget in_=master - see 'in' option description
ipadx=amount - add internal padding in x direction
ipady=amount - add internal padding in y direction
padx=amount - add padding in x direction
pady=amount - add padding in y direction
row=number - use cell identified with given row (starting with 0)
rowspan=number - this widget will span several rows
sticky=NSEW - if cell is larger on which sides will this

widget stick to the cell boundary

`grid_forget()`

Unmap this widget.

`grid_info()`

Return information about the options for positioning this widget in a grid.

`grid_location(x, y)`

Return a tuple of column and row which identify the cell at which the pixel at position X and Y inside the master widget is located.

`grid_propagate(flag=['_noarg_'])`

Set or get the status for propagation of geometry information.

A boolean argument specifies whether the geometry information of the slaves will determine the size of this widget. If no argument is given, the current setting will be returned.

`grid_remove()`

Unmap this widget but remember the grid options.

`grid_rowconfigure(index, cnf={}, **kw)`

Configure row INDEX of a grid.

Valid resources are minsize (minimum size of the row), weight (how much does additional space propagate to this row) and pad (how much space to let additionally).

`grid_size()`

Return a tuple of the number of column and rows in the grid.

`grid_slaves(row=None, column=None)`

Return a list of all slaves of this widget in its packing order.

`image_names()`

Return a list of all existing image names.

`image_types()`

Return a list of all available image types (e.g. photo bitmap).

`info()`

Return information about the packing options for this widget.

`is_data_selected_from_fields()`

`keys()`

Return a list of all resource names of this widget.

`lift(aboveThis=None)`

Raise this widget in the stacking order.

`listBox2 = None`

`listbox = None`

`location(x, y)`

Return a tuple of column and row which identify the cell at which the pixel at position X and Y inside the master widget is located.

`lower(belowThis=None)`

Lower this widget in the stacking order.

`mainloop(n=0)`

Call the mainloop of Tk.

`nametowidget(name)`

Return the Tkinter instance of a widget identified by its Tcl name NAME.

`option_add(pattern, value, priority=None)`

Set a VALUE (second parameter) for an option PATTERN (first parameter).

An optional third parameter gives the numeric priority (defaults to 80).

`option_clear()`

Clear the option database.

It will be reloaded if option_add is called.

`option_get(name, className)`

Return the value for an option NAME for this widget with CLASSNAME.

Values with higher priority override lower values.

`option_readfile(fileName, priority=None)`

Read file FILENAME into the option database.

An optional second parameter gives the numeric priority.

`pack(cnf={}**kw)`

Pack a widget in the parent widget. Use as options: after=widget - pack it after you have packed widget anchor=NSEW (or subset) - position widget according to

given direction

before=widget - pack it before you will pack widget expand=bool - expand widget if parent size grows fill=NONE or X or Y or BOTH - fill widget if widget grows in=master - use master to contain this widget in_=master - see 'in' option description ipadx=amount - add internal padding in x direction ipady=amount - add internal padding in y direction padx=amount - add padding in x direction pady=amount - add padding in y direction side=TOP or BOTTOM or LEFT or RIGHT - where to add this widget.

`pack_configure(cnf={}**kw)`

Pack a widget in the parent widget. Use as options: after=widget - pack it after you have packed widget anchor=NSEW (or subset) - position widget according to

given direction

before=widget - pack it before you will pack widget expand=bool - expand widget if parent size grows fill=NONE or X or Y or BOTH - fill widget if widget grows in=master - use master to contain this

widget in_=master - see 'in' option description ipadx=amount - add internal padding in x direction ipady=amount - add internal padding in y direction padx=amount - add padding in x direction pady=amount - add padding in y direction side=TOP or BOTTOM or LEFT or RIGHT - where to add this widget.

pack_forget()

Unmap this widget and do not use it for the packing order.

pack_info()

Return information about the packing options for this widget.

pack_propagate(*flag*=['noarg_'])

Set or get the status for propagation of geometry information.

A boolean argument specifies whether the geometry information of the slaves will determine the size of this widget. If no argument is given the current setting will be returned.

pack_slaves()

Return a list of all slaves of this widget in its packing order.

place(*cnf*={}, ***kw*)

Place a widget in the parent widget. Use as options: in=master - master relative to which the widget is placed in_=master - see 'in' option description x=amount - locate anchor of this widget at position x of master y=amount - locate anchor of this widget at position y of master relx=amount - locate anchor of this widget between 0.0 and 1.0

relative to width of master (1.0 is right edge)

rely=amount - locate anchor of this widget between 0.0 and 1.0

relative to height of master (1.0 is bottom edge)

anchor=NSEW (or subset) - position anchor according to given direction width=amount - width of this widget in pixel

height=amount - height of this widget in pixel relwidth=amount - width of this widget between 0.0 and 1.0

relative to width of master (1.0 is the same width as the master)

relheight=amount - height of this widget between 0.0 and 1.0
relative to height of master (1.0 is the same height as the master)

bordermode="inside" or "outside" - whether to take border width of master widget into account

place_configure(*cnf*={}, ***kw*)

Place a widget in the parent widget. Use as options: in=master - master relative to which the widget is placed in_=master - see 'in' option description x=amount - locate anchor of this widget at position x of master y=amount - locate anchor of this widget at position y of master relx=amount - locate anchor of this widget between 0.0 and 1.0

relative to width of master (1.0 is right edge)

rely=amount - locate anchor of this widget between 0.0 and 1.0
relative to height of master (1.0 is bottom edge)

anchor=NSEW (or subset) - position anchor according to given direction width=amount - width of this widget in pixel height=amount - height of this widget in pixel relwidth=amount - width of this widget between 0.0 and 1.0

relative to width of master (1.0 is the same width as the master)

relheight=amount - height of this widget between 0.0 and 1.0
relative to height of master (1.0 is the same height as the master)

bordermode="inside" or "outside" - whether to take border width of master widget into account

place_forget()

Unmap this widget.

`place_info()`

Return information about the placing options for this widget.

`place_slaves()`

Return a list of all slaves of this widget in its packing order.

`populate_object(dirPath)`

`propagate(flag=['noarg'])`

Set or get the status for propagation of geometry information.

A boolean argument specifies whether the geometry information of the slaves will determine the size of this widget. If no argument is given the current setting will be returned.

`quit()`

Quit the Tcl interpreter. All widgets will be destroyed.

`register(func, subst=None, needcleanup=1)`

Return a newly created Tcl function. If this function is called, the Python function FUNC will be executed. An optional function SUBST can be given which will be executed before FUNC.

`rowconfigure(index, cnf={}**kw)`

Configure row INDEX of a grid.

Valid resources are minsize (minimum size of the row), weight (how much does additional space propagate to this row) and pad (how much space to let additionally).

`selection_clear(**kw)`

Clear the current X selection.

`selection_get(**kw)`

Return the contents of the current X selection.

A keyword parameter selection specifies the name of the selection and defaults to PRIMARY. A keyword parameter displayof specifies a widget on the display to use. A keyword parameter type specifies the form of data to be fetched, defaulting to STRING except on X11, where UTF8_STRING is tried before STRING.

`selection_handle(command, **kw)`

Specify a function COMMAND to call if the X selection owned by this widget is queried by another application.

This function must return the contents of the selection. The function will be called with the arguments OFFSET and LENGTH which allows the chunking of very long selections. The following keyword parameters can be provided: selection - name of the selection (default PRIMARY), type - type of the selection (e.g. STRING, FILE_NAME).

`selection_own(**kw)`

Become owner of X selection.

A keyword parameter selection specifies the name of the selection (default PRIMARY).

`selection_own_get(**kw)`

Return owner of X selection.

The following keyword parameter can be provided: selection - name of the selection (default PRIMARY), type - type of the selection (e.g. STRING, FILE_NAME).

`send(interp, cmd, *args)`

Send Tcl command CMD to different interpreter INTERP to be executed.

`setvar(name='PY_VAR', value='I')`

Set Tcl variable NAME to VALUE.

size()

Return a tuple of the number of column and rows in the grid.

slaves()

Return a list of all slaves of this widget in its packing order.

tk_bisque()

Change the color scheme to light brown as used in Tk 3.6 and before.

tk_focusFollowsMouse()

The widget under mouse will get automatically focus. Can not be disabled easily.

tk_focusNext()

Return the next widget in the focus order which follows widget which has currently the focus.

The focus order first goes to the next child, then to the children of the child recursively and then to the next sibling which is higher in the stacking order. A widget is omitted if it has the takefocus resource set to 0.

tk_focusPrev()

Return previous widget in the focus order. See tk_focusNext for details.

tk_setPalette(*args, **kw)

Set a new color scheme for all widget elements.

A single color as argument will cause that all colors of Tk widget elements are derived from this. Alternatively several keyword parameters and its associated colors can be given. The following keywords are valid: activeBackground, foreground, selectColor,

activeForeground, highlightBackground, selectBackground, background, highlightColor, selectForeground, disabledForeground, insertBackground, troughColor.

tk_strictMotif(*boolean=None*)

Set Tcl internal variable, whether the look and feel should adhere to Motif.

A parameter of 1 means adhere to Motif (e.g. no color change if mouse passes over slider). Returns the set value.

tkraise(*aboveThis=None*)

Raise this widget in the stacking order.

unbind(*sequence, funcid=None*)

Bind for this widget for event SEQUENCE the function identified with FUNCID.

unbind_all(*sequence*)

Bind for all widgets for event SEQUENCE all functions.

unbind_class(*className, sequence*)

Bind for all widgets with bindtag CLASSNAME for event SEQUENCE all functions.

update()

Enter event loop until all pending events have been processed by Tcl.

update_idletasks()

Enter event loop until all idle callbacks have been called. This will update the display of windows but not process events caused by the user.

wait_variable(*name='PY_VAR'*)

Wait until the variable is modified.

A parameter of type IntVar, StringVar, DoubleVar or BooleanVar must be given.

`wait_visibility(window=None)`

Wait until the visibility of a WIDGET changes (e.g. it appears).

If no parameter is given self is used.

`wait_window(window=None)`

Wait until a WIDGET is destroyed.

If no parameter is given self is used.

`waitvar(name='PY_VAR')`

Wait until the variable is modified.

A parameter of type IntVar, StringVar, DoubleVar or BooleanVar must be given.

`winfo_atom(name, displayof=0)`

Return integer which represents atom NAME.

`winfo_atomname(id, displayof=0)`

Return name of atom with identifier ID.

`winfo_cells()`

Return number of cells in the colormap for this widget.

`winfo_children()`

Return a list of all widgets which are children of this widget.

`winfo_class()`

Return window class name of this widget.

`winfo_colormapfull()`

Return True if at the last color request the colormap was full.

`winfo_containing(rootX, rootY, displayof=0)`

Return the widget which is at the root coordinates ROOTX, ROOTY.

`winfo_depth()`

Return the number of bits per pixel.

`winfo_exists()`

Return true if this widget exists.

`winfo_fpixels(number)`

Return the number of pixels for the given distance NUMBER (e.g. "3c") as float.

`winfo_geometry()`

Return geometry string for this widget in the form "widthxheight+X+Y".

`winfo_height()`

Return height of this widget.

`winfo_id()`

Return identifier ID for this widget.

`winfo_interps(displayof=0)`

Return the name of all Tcl interpreters for this display.

`winfo_ismapped()`

Return true if this widget is mapped.

`winfo_manager()`

Return the window manager name for this widget.

`winfo_name()`

Return the name of this widget.

winfo_parent()

Return the name of the parent of this widget.

winfo_pathname(*id*, *displayof*=0)

Return the pathname of the widget given by ID.

winfo_pixels(*number*)

Rounded integer value of winfo_fpixels.

winfo_pointerx()

Return the x coordinate of the pointer on the root window.

winfo_pointery()

Return a tuple of x and y coordinates of the pointer on the root window.

winfo_pointery()

Return the y coordinate of the pointer on the root window.

winfo_reqheight()

Return requested height of this widget.

winfo_reqwidth()

Return requested width of this widget.

winfo_rgb(*color*)

Return a tuple of integer RGB values in range(65536) for color in this widget.

winfo_rootx()

Return x coordinate of upper left corner of this widget on the root window.

winfo_rooty()

Return y coordinate of upper left corner of this widget on the root window.

winfo_screen()

Return the screen name of this widget.

winfo_screencells()

Return the number of the cells in the colormap of the screen of this widget.

winfo_screendepth()

Return the number of bits per pixel of the root window of the screen of this widget.

winfo_screenheight()

Return the number of pixels of the height of the screen of this widget in pixel.

winfo_screenmmheight()

Return the number of pixels of the height of the screen of this widget in mm.

winfo_screenmmwidth()

Return the number of pixels of the width of the screen of this widget in mm.

winfo_screenvisual()

Return one of the strings directcolor, grayscale, pseudocolor, staticcolor, staticgray, or truecolor for the default colormodel of this screen.

winfo_screenwidth()

Return the number of pixels of the width of the screen of this widget in pixel.

winfo_server()

Return information of the X-Server of the screen of this widget in the form "XmajorRminor vendor vendorVersion".

winfo_toplevel()

Return the toplevel widget of this widget.

winfo_viewable()

Return true if the widget and all its higher ancestors are mapped.

winfo_visual()

Return one of the strings directcolor, grayscale, pseudocolor, staticcolor, staticgray, or truecolor for the colormodel of this widget.

winfo_visualid()

Return the X identifier for the visual for this widget.

winfo_visualsavailable(*includeids=False*)

Return a list of all visuals available for the screen of this widget.

Each item in the list consists of a visual name (see winfo_visual), a depth and if includeids is true is given also the X identifier.

winfo_vrootheight()

Return the height of the virtual root window associated with this widget in pixels. If there is no virtual root window return the height of the screen.

winfo_vrootwidth()

Return the width of the virtual root window associated with this widget in pixel. If there is no virtual root window return the width of the screen.

winfo_vrootx()

Return the x offset of the virtual root relative to the root window of the screen of this widget.

winfo_vrooty()

Return the y offset of the virtual root relative to the root window of the screen of this widget.

winfo_width()

Return the width of this widget.

winfo_x()

Return the x coordinate of the upper left corner of this widget in the parent.

winfo_y()

Return the y coordinate of the upper left corner of this widget in the parent.

DefragHistoryGrapher.py

This module contains the basic functions for plotting DefragHistory time series generated from a configuration file. To run this script you must use a GUI, as pyplot requires the environment variable \$DISPLAY to be set to generate the PDF files.

Args:

--outfile: String for the name of the output file (without the suffix) --
inputFile: String of the path name for the configuration file containing the data for the time-series --numCores: Integer for the number of cores from which data was pulled --mode: Integer value for run mode (1=ADP, 2=CDR) --encapStruct: String for the name of the struct where the set points are contained (ADP) --trackingNames: comma-separated strings for the names of the stats to be tracked in the main axis of the graph --secondaryVars: comma-separated strings for the names of the fields to be plotted in the secondary axis --bandwidthFlag: Boolean flag that indicates if the secondary axis corresponds to bandwidth --
headerName: String for the name of the field associated with the header (ADP) --indexName: String for the name of the field associated with the index --timeLabel: String for the name of the field associated with time --coreLabel: String for the name of the struct associated with core

number --logTypeLabel: String for the name of the field associated with the log type (ADP) --setPointNames: Comma-separated strings for the names of the set points to be used in the graph (ADP) --setPointValues: Comma-separated integers for the values of the set points to be used in the graph (CDR) --arbiterLabels: Comma-separated strings for the names of the set points contained in the arbiter structure (CDR) --colors: Comma-separated strings of the color names for the set points to be used in graph (ADP) --debug: Boolean flag to activate debug statements

Example:

Default usage:

```
$ python DefragHistoryGapher.py
```

Specific usage:

```
$ python DefragHistoryGapher.py --outfile test1 --inputFile time-series/time-series.ini --numCores 2  
--debug True
```

class

```
src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher(mode=1,  
encapsulatingStruct='__anuniontypedef115__', trackingNames=None,  
secondaryVars=None, bandwidthFlag=True, headerName='header',  
indexName='index', timeLabel='time', coreLabel='core',  
logTypeLabel='prevlogtype', setPointNames=None, setPointValues=None,  
hostOperations=None, ctxList=None, NandOperations=None,  
slotList=None, colors=None, arbiterLabels=None, debug=False)
```

Bases: `object`

Class for graphing DefragHistory for both ADP and CDR drives

Attributes:

debug: Boolean flag to activate debug statements mode: Integer value for run mode (1=ADP, 2=CDR) defaultColors: List of strings for the names of standard colors to be used for plotting tracked and secondary variables encapsulatingStruct: String for the name of the struct where the set points are contained (ADP) setPointNames: List of strings for the names of the set points to be used in the graph

(ADP) trackingNames: List of strings for the names of the fields to be graphed in the main axis secondaryVars: List of strings for the names of the fields to be graphed in the secondary axis bandwidthFlag: Boolean flag that indicates if the secondary axis corresponds to bandwidth headerName: String for the name of the field associated with the header (ADP) indexName: String for the name of the field associated with the index logTypeLabel: String for the name of the field associated with the log type (ADP) timeLabel: String for the name of the field associated with time coreLabel: String for the name of the struct associated with core number colorDict: Dictionary of Set Point Names to string names for colors setPointDict: Dictionary of Set Point Names to integer values arbiterDict: Dictionary of Set Point Names to string names for the arbiter fields

`DefragHistoryGrapherAPI(input_t: str = 'time-series.ini', out: str = 'telemetryDefault', numCores: int = 2, bandwidthFlag: bool = True)`

API to replace standard command line call (the visualizeTS class has to instantiated before calling this method)

Args:

input_t: String of the name for the configuration file containing the data for the time-series
out: String of the prefix for the output file
numCores: Integer for the number of cores from which data was pulled
bandwidthFlag: Boolean flag that indicates if the secondary axis corresponds to bandwidth

Returns:

`class defragHistoryUtility`

Bases: `object`

`static calculateBandwidth(currentTime, currentCore, currentData, field, coresBandwidth, seriesDeltaClass)`

function for calculating and appending the bandwidth value to the value list for a field

Args:

currentTime: Integer value for the time stamp collected in the dataDict
currentCore: Integer value for the current core number being accessed
currentData: Integer for the current data value being accessed
field: String for the name of the field to be accessed
coresBandwidth: Dictionary for the values to be plotted in the secondary axis
seriesDeltaClass: Child class of Series containing fields for previous time and data

Returns:

static generateTruncatedLists(coreElements: dict, field, start: int = 0, end: int = 100)

function for generating the time and data lists of the corresponding percentage of values

Args:

coreElements: Series dictionary containing the data and time values
field: String for the name of the field to be accessed in the Series dictionary
start: Integer value for the start percentage of the lists
end: Integer value for the end percentage of the lists

Returns:

timeList: list containing the time values for the corresponding percentages
dataList: list containing the data values for the corresponding percentages

static getMaxAIU(mode, setPointNames, coreSetPoints, fieldData)

function for calculating the maximum AIU to be used in the graphs for both CDR and ADP

Args:

mode: Integer value for run mode (1=ADP, 2=CDR)
setPointNames: List of strings for the names of the set points to be used in the graph
coreSetPoints: Dictionary of set point

names to series (ADP) or lists (CDR) of data values
fieldData: List of data values for current field

Returns:

Float for the maximum value for AIU to be used in the graph

static initializeDataDicts(numCores=2, targetDict=None, defaultDict=None, manualSetPoints=False)

function for initializing the intermediate dictionaries that contain the data series to be plotted

Args:

numCores: Integer for the number of cores from which data was pulled
targetDict: Dictionary where the values of the setPoints will be stored
defaultDict: Dictionary with the default values for setPoints
manualSetPoints: Boolean flag to indicate if the default set points should be used

Returns:

coresDict: Dictionary of data values for the fields to be plotted in the main axis
coresSecondaryDict: Dictionary with the data values for the fields to be plotted in the secondary axis
previousLogNumDict: Dictionary with the previous log number to be used as a sentinel value

static populateEncapsulatedField(index, currentTime, currentCore, field, encapStruct, logDict, coresDict, seriesClass)

function for populating the field inside coresDict with the respective time and data

Args:

encapStruct: Constant auto parse detection module.
index: Integer for the position in the value list
currentTime: Integer value for the time stamp collected in the dataDict
currentCore: Integer value for the current core number being accessed
field: String for the name of the field to be accessed
logDict: Dictionary associated with the current log

coresDict: Dictionary of data values for the fields to be plotted in the main axis seriesClass: Class containing lists for time and data

Returns:

static populateField(index, currentTime, currentCore, field, logDict, coresDict, seriesClass)

function for populating the field inside coresDict with the respective time and data

Args:

index: Integer for the position in the value list
currentTime: Integer value for the time stamp collected in the dataDict
currentCore: Integer value for the current core number being accessed
field: String for the name of the field to be accessed
logDict: Dictionary associated with the current log
coresDict: Dictionary of data values for the fields to be plotted in the main axis seriesClass: Class containing lists for time and data

Returns:

generateDataDictFromConfig(input_t)

function for reading the config file into an intermediate dict, and transforming that intermediate dict to produce the nesting of the original object

Args:

input_t: String of the name for the configuration file containing the data for the time-series

Returns:

Dictionary containing all the time-series data with the appropriate nesting according to the fields

```
generateTSVisualizationADP(object_t, dataDict: Optional[dict] =  
    None, bandwidthFlag: bool = True, numCores: int = 2, pp=None,  
    start=0, end=100)
```

function for generating a line plot for the time series data collected from an ADP drive

Args:

object_t: String for the object identifier (ex. uid-6) *dataDict*: Dictionary containing all the time-series data with the appropriate nesting *bandwidthFlag*: Boolean flag that indicates if the secondary axis corresponds to bandwidth *numCores*: Integer for the number of cores from which data was pulled *pp*: File descriptor for the PDF file where the plots will be saved *start*: Integer value for the start percentage of the lists to be graphed *end*: Integer value for the end percentage of the lists to be graphed

Returns:

```
generateTSVisualizationADPinPDF(object_t, dataDict: Optional[dict] =None, bandwidthFlag: bool = True, numCores: int = 2,  
pdfFiles=None, start=0, end=100)
```

function for generating a line plot for the time series data collected from an ADP drive

Args:

object_t: String for the object identifier (ex. uid-6) *dataDict*: Dictionary containing all the time-series data with the appropriate nesting *bandwidthFlag*: Boolean flag that indicates if the secondary axis corresponds to bandwidth *numCores*: Integer for the number of cores from which data was pulled *pdfFiles*: list of PDF files in which the figures will be saved *start*: Integer value for the start percentage of the lists to be graphed *end*: Integer value for the end percentage of the lists to be graphed

Returns:

`generateTSVisualizationCDR(object_t, dataDict, bandwidthFlag,
numCores=2, pp=None, start=0, end=100)`

function for generating a line plot for the time series of
DefragHistory for a CDR drive

Args:

object_t: String for the object identifier (ex. uid-6) *dataDict*: Dictionary containing all the time-series data with the appropriate nesting *bandwidthFlag*: Boolean flag that indicates if the secondary axis corresponds to bandwidth *numCores*: Integer for the number of cores from which data was pulled *pp*: File descriptor for the PDF file where the plots will be saved *start*: Integer value for the start percentage of the lists to be graphed *end*: Integer value for the end percentage of the lists to be graphed

Returns:

`generateTSVisualizationCDRinPDF(object_t, dataDict, bandwidthFlag,
numCores=2, pdfFiles=None, start=0, end=100)`

function for generating a line plot for the time series of
DefragHistory for a CDR drive

Args:

object_t: String for the object identifier (ex. uid-6) *dataDict*: Dictionary containing all the time-series data with the appropriate nesting *bandwidthFlag*: Boolean flag that indicates if the secondary axis corresponds to bandwidth *numCores*: Integer for the number of cores from which data was pulled *pdfFiles*: list of pdf files in which the graphs will be stored *start*: Integer value for the start percentage of the lists to be graphed *end*: Integer value for the end percentage of the lists to be graphed

Returns:

`getDebug()`

function for reading the debug flag stored in the DefragHistoryGrapher attributes

Returns:

Boolean flag to activate debug statements

getSecondaryVars()

function for reading the list secondaryVars stored in the DefragHistoryGrapher attributes

Returns:

List of strings with the names for the variables to be plotted in the secondary axis

getSetPointNames()

function for reading the list newSetPointNames stored in the DefragHistoryGrapher attributes

Returns:

List of strings for the names of the set points

getSetPointsCDR(*dataDict*, *numCores*, *dataSetLength*)

function for obtaining the set point values from the arbiter object for a CDR drive

Args:

dataDict: Dictionary containing all the objects from the configuration file
numCores: Integer for the number of cores from which data was pulled
dataSetLength: How many data points are in each field (EX: if 100 samples were taken for 2 cores, *dataSetLength* will be 200)

Returns:

setPointDict: Dictionary of set point values for each core colors:
Dictionary of colors for data set point names

getTrackingNames()

function for reading the list trackingNames stored in the DefragHistoryGrapher attributes

Returns:

List of strings with the names for the variables to be plotted in the main axis

`graphTSVisualizationADP(numCores, coresDict, coresSecondaryDict, bandwidthFlag, pdf=None, start=0, end=100)`

function for creating and saving the time-series plot for available blocks and bandwidth

Args:

numCores: Integer for the number of cores from which data was collected in the configuration file
coresDict: Dictionary of data values for the fields to be plotted in the main axis

coresSecondaryDict: Dictionary with the data values for the fields to be plotted in the secondary axis

bandwidthFlag: Boolean flag that indicates if the secondary axis corresponds to bandwidth

pdf: File descriptor for the PDF file where the plots will be saved

start: Integer value for the start percentage of the lists to be graphed

end: Integer value for the end percentage of the lists to be graphed

Returns:

`graphTSVisualizationADPinPDF(coreNumber, coresDict, coresSecondaryDict, bandwidthFlag, pdf=None, start=0, end=100)`

function for creating and saving the time-series plot for available blocks and bandwidth

Args:

numCores: Integer for the number of cores from which data was collected in the configuration file
coresDict: Dictionary of data values for the fields to be plotted in the main axis

coresSecondaryDict: Dictionary with the data values for the fields to be plotted in the secondary axis

bandwidthFlag:

Boolean flag that indicates if the secondary axis corresponds to bandwidth pdf: File descriptor for the PDF file where the plots will be saved start: Integer value for the start percentage of the lists to be graphed end: Integer value for the end percentage of the lists to be graphed

Returns:

`graphTSVisualizationCDR(numCores, coresDict, setPointsDict, colors, coresSecondaryDict, bandwidthFlag, pdf=None, start=0, end=100)`
function for creating and saving the time-series plot for available blocks and bandwidth

Args:

numCores: Integer for the number of cores from which data was collected in the configuration file
coresDict: Dictionary data values for the fields to be plotted in the main axis
setPointsDict: Dictionary of set point values for each core
colors: Dictionary of colors for data set point names
coresSecondaryDict: Dictionary with the data values for the fields to be plotted in the secondary axis
bandwidthFlag: Boolean flag that indicates if the secondary axis corresponds to bandwidth
pdf: File descriptor for the PDF file where the plots will be saved
start: Integer value for the start percentage of the lists to be graphed
end: Integer value for the end percentage of the lists to be graphed

Returns:

`graphTSVisualizationCDRinPDF(coreNumber, coresDict, setPointsDict, colors, coresSecondaryDict, bandwidthFlag, pdf=None, start=0, end=100)`
function for creating and saving the time-series plot for available blocks and bandwidth

Args:

coreNumber: number of the core to be graphed
coresDict: Dictionary data values for the fields to be plotted in the main

axis setPointsDict: Dictionary of set point values for each core
colors: Dictionary of colors for data set point names
coresSecondaryDict: Dictionary with the data values for the fields to be plotted in the secondary axis bandwidthFlag: Boolean flag that indicates if the secondary axis corresponds to bandwidth pdf: File descriptor for the PDF file where the plots will be saved start: Integer value for the start percentage of the lists to be graphed end: Integer value for the end percentage of the lists to be graphed

Returns:

`populateTSDATAADP(object_t, numCores: int = 2, dataDict:
Optional[dict] = None, bandwidthFlag: bool = True,
encapStruct='anuniontypedef46')`

@todo dgarces these should be dynamic based on the log type. The current code uses exceptions to bypass not found terms; however, these should be handled by the log type. function for filling the intermediate time-series data structures with the ADP data contained inside the given dictionary

Args:

encapStruct: Contant auto parse detection variable. *object_t*: String for the object identifier (ex. uid-6) *numCores*: Integer for the number of cores from which data was pulled *dataDict*: Dictionary containing all the time-series data with the appropriate nesting *bandwidthFlag*: Boolean flag that indicates if the secondary axis corresponds to bandwidth

Returns:

coresDict: Dictionary of data values for the fields to be plotted in the main axis *coresSecondaryDict*: Dictionary with the data values for the fields to be plotted in the secondary axis

`populateTSDATACDR(object_t, numCores, dataDict, bandwidthFlag)`

function for filling the intermediate time-series data structures with the CDR data contained inside the given dictionary

Args:

object_t: String for the object identifier (ex. uid-6) numCores: Integer for the number of cores from which data was collected in the configuration file dataDict: Dictionary containing all the time-series data with the appropriate nesting bandwidthFlag: Boolean flag that indicates if the secondary axis corresponds to bandwidth

Returns:

setPointsDict: Dictionary of set point values for each core colors: Dictionary of colors for data set point names coresDict: Dictionary of data values for the fields to be plotted in the main axis coresSecondaryDict: Dictionary with the data values for the fields to be plotted in the secondary axis

`setDebug(debug)`

function for setting the debug flag stored in the DefragHistoryGrapher attributes

Args:

debug: Boolean flag to activate debug statements

Returns:

`setSecondaryVars(secondaryVars)`

function for setting the list secondaryVars stored in the DefragHistoryGrapher attributes

Args:

secondaryVars: list of strings with the names for the variables to be plotted in the secondary axis

Returns:

`setSetPointNames(newSetPointNames)`

function for setting the list newSetPointNames stored in the DefragHistoryGrapher attributes

Args:

newSetPointNames: list of strings for the names of the set points

Returns:

setTrackingNames(*trackingNames*)

function for setting the list trackingNames stored in the DefragHistoryGrapher attributes

Args:

trackingNames: list of strings with the names for the variables to be plotted in the main axis

Returns:

writeTSVisualizationToPDF(*object_t*: str = 'uid-41', *dataDict*: Optional[dict] = None, *bandwidthFlag*: bool = True, *out*: str = '', *numCores*: int = 2, *start*=0, *end*=100)

function that generates the time series visualization for DefragHistory and saves it into a PDF file named with the prefix defined by out followed by a lower dash and the uid of the object

Args:

start: Starting point of time series
end: end point of time series
dataDict: Dictionary containing all the time-series data with the appropriate nesting
bandwidthFlag: Boolean flag that indicates if the secondary axis corresponds to bandwidth
out: String of the prefix for the output file
numCores: Integer for the number of cores from which data was pulled

Returns:

src.software.TSV.DefragHistoryGrapher.main()

main function to be called when the script is directly executed from the command line

utilityTSV.py

class src.software.TSV.utilityTSV.utilityTSV

Bases: **object**

static checkBinDir(binDir)

function to set the default value for binDir

Args:

binDir: path to the root directory containing the bin directories for the time series

Returns:

String representation of the path to the root bin directory

static checkBooleanOption(booleanOption)

function to set the default value for a boolean option to True and turn it from a string into a boolean value

Args:

booleanOption: string representation of boolean value for boolean flag

Returns:

Boolean value for boolean flag

static checkCombine(combine)

function to set the default value for combine and turn it from a string into a boolean value

Args:

combine: string representation of boolean value for combine flag

Returns:

Boolean value for combine flag

static checkDebugOption(debugOptions)

function to set the default value for debugOptions and turn it from a string into a boolean value

Args:

 debugOptions: string representation of boolean value for debug flag

Returns:

 Boolean value for debug flag

static checkDriveName(driveName)

function to set the default value for driveName for the nvme-cli interface

Args:

 driveName: Name of device interface to get data from

Returns:

 String representation of the drive name

static checkDriveNumber(driveNumber)

function to set the default value for driveNumber and turn it from a string into an int

Args:

 driveNumber: string representation of int for drive number

Returns:

 Int value for drive number

static checkFWDDir(fwDir)

function to set the default value for fwDir

Args:

 fwDir: path to the firmware build directory containing the python parsers

Returns:

 String representation of the path to the firmware build directory

static checkIdentifier(identifier)

function to set the default value for identifier

Args:

identifier: String for the name of the data set that corresponds to the telemetry pull to be executed

Returns:

String for the name of the data set

static checkIniOutfile(outfile)

function to set the default value for outfile

Args:

outfile: name for the output file where the configs will be stored

Returns:

String value for the name of the output file

static checkInputDir(inputDir)

function to set the default value for inputDir

Args:

inputDir: path to the directory where the binaries are stored

Returns:

String representation of the path to the input directory

static checkInputFile(inputFile)

function to set the default value for inputFile

Args:

inputFile: Path of the file containing the config that describes the time series

Returns:

String representation of the path to the file containing the config for the time series

*static checkIterations(*iterations*)*

function to set the default value for iterations and turn it from a string into an int

Args:

iterations: string representation of int value for the number of iterations

Returns:

Int value for number of iterations

*static checkModeSelect(*modeSelect*)*

function to set the default value for modeSelect and turn it from a string into an int

Args:

modeSelect: string representation of int value for modeSelect flag

Returns:

Int value for modeSelect flag

*static checkNumCores(*numCores*)*

function to set the default value for numCores and turn it from a string into an int

Args:

numCores: string representation of int value for the number of cores from which data was pulled

Returns:

Int value for number of cores

*static checkOutfile(*outfile*)*

function to set the default value for outfile

Args:

outfile: Name for the output file where the visualizations will be stored

Returns:

String representation of the name for the output file (without suffix)

static checkOutpath(outpath)

function to set the default value for outfile

Args:

outpath: path for the output directory where the intermediate and output files will be stored

Returns:

String value for the path of the output directory

static checkOutputDir(outputDir)

function to set the default value for outputDir

Args:

outputDir: path to the directory where the binaries will be stored

Returns:

String representation of the path to the output directory

static checkParseOption(parseFlag)

function to set the default value for parse flag and turn it from a string into a boolean value

Args:

parseFlag: string representation of boolean value for parse flag

Returns:

Boolean value for parse flag

static checkSubSeqLen(subSeqLen)

function to set the default value for subSeqLen and turn it from a string into an integer value

Args:

subSeqLen: string representation of integer value for length of sliding window for Matrix profile

Returns:

Integer value for the sliding window length

static checkTransformTS(transformTS)

function to set the default value for transformTS and turn it from a string into a boolean value

Args:

transformTS: string representation of boolean value for transformTS flag

Returns:

Boolean value for transformTS flag

static getDateFromName(dirName='sample', suffix='.bin')

function to extract datetime object from directory name

Args:

dirName: String representation of the directory name

Returns:

datetime object containing the date parsed in the dirName

configToText.py

This module contains the basic functions for turning a configuration file into multiple plain text files that each contain a single object structure to be turned back again into its binary representation. The plain text files have .txt suffix, while the configuration file has .ini suffix

Args:

--inputFile: String of the name for the configuration file containing the data for the time-series
--iterations: Integer number of data points to be considered in the time series
--identifier: String for the name of the data set that corresponds to the telemetry pull to be executed
--debug: Boolean flag to activate debug statements

Example:

Default usage:

```
$ python configToText.py
```

Specific usage:

```
$ python configToText.py --inputFile time-series.ini --iterations 100  
--identifier Tv2HiTAC --debug True
```

class src.software.TSV.configToText.configToText(debug=False)

Bases: `object`

*generateTextFromConfigFile(inputFile='time-series.ini',
identifier='Tv2HiTAC', iterations=100)*

function for turning a configuration file into multiple plain text files, each containing all the objects for a single time stamp in the time-series of the configuration file

Args:

inputFile: String of the path to the configuration file to be used
for processing identifier: String of the identifier for the name of the plain text files

Returns:

generateTextFromDict(tempDict, accumStr, indentLevel, index)

Recursive function to process a single entry of the expanded dictionary and turn it into a plain text line

Args:

tempDict: Expanded dictionary containing the fields for an object
accumStr: String storing the lines processed so far
indentLevel: Integer indicating the nesting level of the current line
index: Integer for the index of the current stamp in the time-series

Returns:

String of the lines processed so far

`getDebug()`

function for reading the debug flag stored in the visualizeTS attributes

Returns:

Boolean flag to activate debug statements

`setDebug(debug)`

function for setting the debug flag stored in the visualizeTS attributes

Args:

`debug`: Boolean flag to activate debug statements

Returns:

`src.software.TSV.configToText.main()`

main function to be called when the script is directly executed from the command line

`formatTSFiles.py`

This module contains the basic functions for generating the plain text files and the ini files for a time series. The plain text files have .txt suffix, while the ini files have .ini suffix

Args:

`--outfile`: String for the name of the output file (must contain the .ini suffix)
`--outpath`: String for the path of the output directory where the

intermediate files will be stored
--targetObject: String for the object name to be processed
--fwDir: String for the path to the firmware build directory containing the python parsers
--binDir: String for the path to the root directory containing the bin directories for the time series
--mode: Integer value for run mode (1=bufDict, 2=autoParsers)
--debug: Boolean flag to activate debug statements

Example:

Default usage:

```
$ python formatTSFiles.py
```

Specific usage:

```
$ python formatTSFiles.py --outfile time-series.ini --outpath ./time-series --targetObject ThermalSensor  
--fwDir ../projects/arbordaleplus_t2 --binDir ./binaries --debug  
True
```

*class src.software.TSV.formatTSFiles.formatTSFiles(*outpath='sample'*,
debug=False)*

Bases: **object**

debug = False

*digestTextFiles(*plainTextFiles=None*, *nlogFiles=None*, *outfile='time-series.ini'*, *dataDictionaryDebugFileName=None*, *obj=None*, *mode=1*)*

function to accumulate the telemetry plain text files into a single configuration file to be used for the time series

Args:

plainTextFiles: List of plain text file names to be parsed into the configuration file
outfile: String for the name of the output file (must contain the .ini suffix)
dataDictionaryDebugFileName: debug data dump file for all uid tokens.
obj: String for the name of telemetry objects to be extracted from the plain text files
mode: Integer value for run mode (1=bufDict, 2=autoParsers)

Returns: True if telemetry is valid

`extractNlogTime(line)`

`formatTSFilesAPI(fwDir=None, binDir='binaries', outfile='time-series.ini', obj=None, mode=1, binList=None, nlogFolder=None, timeStamp=None, debugFile=None, debug=None)`

API to replace standard command line call (the `formatTSFiles` class has to instantiated before calling this method)

Args:

`debugFile`: data debug log file for understanding parser info.
 `debug`: developer debug flag. `timeStamp`: file formated time stamp
 `binList`: `fwDir`: String for the path to the firmware build directory containing the python parsers
 `binDir`: String for the path to the root directory containing the bin directories for the time series
 `outfile`: String for the name of the output file (must contain the .ini suffix)
 `obj`: String for the name of telemetry objects to be extracted from the plain text files
 `mode`: Integer value for run mode (1=`buffDict`, 2=`autoParsers`)

Returns:

`class formatUtility(debug=False)`

Bases: `object`

Utility class to be used inside `formatTSFiles`

`debug = False`

`processDict(resultDict)`

function that flattens the dictionary to be written into the csv file

Args:

`resultDict`: Dictionary to be flatten

Returns:

 Flatten dictionary

`processTextFile(openFile, resultDict, obj=None, mode=1)`

function to process a single plain text file into the dictionary structure

Args:

openFile: File descriptor for the open file
resultDict: Dictionary structure that will contain all the objects, fields, and their respective values
obj: String for the name of telemetry objects to be extracted from the plain text file
mode: Integer value for run mode (1=bufDict, 2=autoParsers)

Returns:

`generatePlainText(fwDir=None, binDir='binaries', nlogFolder=None, mode=1, binList=None)`

function to generate plain text files from the telemetry binary files

Args:

fwDir: String for the path to the firmware build directory containing the python parsers
binDir: String for the path to the root directory containing the bin directories for the time series
mode: Integer value for run mode (1=bufDict, 2=autoParsers)

Returns:

A list of plain text files generated from the telemetry binary files

`getDebug()`

function for reading the debug flag stored in the formatTSFiles attributes

Returns:

Boolean flag to activate debug statements

`getOutpath()`

function for reading the outpath stored in the formatTSFiles attributes

Returns:

String for the path of the output directory where the plain text files will be stored

isTelemetryValid(*resultDict*, *threshold*=0.6)

Validate the collected telemetry data. For each data object, Check if each field is non-empty

Args: None

Returns: boolean (true if valid)

processNlogFiles(*nlogFiles*=None, *outfile*='time-series.ini')

Args:

nlogFiles: outfile:

Returns:

processPlainTextFiles(*plainTextFiles*=None, *outfile*='time-series.ini', *dataDictionaryDebugFileName*=None, *obj*=None, *mode*=1)

Args:

plainTextFiles: outfile: dataDictionaryDebugFileName: obj:
mode:

Returns:

setDebug(*debug*)

function for setting the debug flag stored in the formatTSFiles attributes

Args:

debug: Boolean flag to activate debug statements

Returns:

setOutpath(*outpath*)

function for setting the outpath stored in the formatTSFiles attributes

Args:

 outpath: String for the path of the output directory where the plain text files will be stored

Returns:

`src.software.TSV.formatTSFiles.main()`

main function to be called when the script is directly executed from the command line

`generateTSBinaries.py`

This module contains the basic functions for generating the telemetry binary files for a time series. It is assumed that telemetry pulls take approximately the same amount of time and they are evenly spaced during the iterations

Args:

 --driveNumber: Integer value for the number of the drive from which telemetry data will be pulled
 --driveName: String for name of device interface to get data from
 --outputDir: String for the path of the output directory where the binaries will be stored
 --inputDir: String for the path of the input directory where the binaries to be parsed are stored
 --debug: Boolean flag to activate debug statements
 --modeSelect: Integer value for run mode (1=hybrid/nvme-cli, 2=internal, 3=parse-only, 4=intelmas)
 --iterations: Integer number of data points to be considered in the time series
 --identifier: String for the name of the data set that corresponds to the telemetry pull to be executed
 --time: Integer time (in seconds) for which data will be collected

Example:

Default usage:

`$ python generateTSBinaries.py`

Specific usage (for extracting telemetry):

```
$ python generateTSBinaries.py --driveNumber 0 --outputDir sample  
--debug True --modeSelect 2 --iterations 20 --time 6
```

Specific usage (for parsing only):

```
$ python generateTSBinaries.py --outputDir sample --inputDir  
./AllBinaries --debug True --modeSelect 3
```

class

src.software.TSV.generateTSBinaries.generateTSBinaries(*iterations*=500,
outpath=*None*, *debug*=*False*)

Bases: **object**

static GetSortedConfigFiles(inputFolder, suffix='.bin', latest=False)

*generateTSBinariesAPI(mode=3, time=None,
driveName='/dev/nvme0n1', driveNumber=0, identifier='Tv2HiTAC',
inputFolder='input-binaries')*

API to replace standard command line call (the generateTSBinaries class has to instantiated before calling this method)

Args:

mode: Integer value for run mode (1=hybrid/nvme-cli, 2=internal, 3=parse-only)
time: Integer time (in seconds) for which data will be collected
driveName: String for name of device interface to get data from
driveNumber: Integer value for the number of the drive from which telemetry data will be pulled
identifier: String for the name of the data set that corresponds to the telemetry pull to be executed
inputFolder: String of the relative path to the directory containing all the binary files to be parsed

Returns:

class generateUtility

Bases: **object**

Utility class to be used inside generateTSBinaries

*IntelIMASCommand(tool='intelmas', operation='dump',
outputFile='telemetry_data.bin', driveNumber=1)*

function for generating a telemetry log using the intelmas tool commands

Args:

tool: String representation of the tool set to be used for the telemetry pull operation: String representation of the operation to be performed (refer to the notes below or the

documentation for the tool chosen)

outputFile: String of the name for the output file

driveNumber: Integer value for the drive ID number

Returns:

Notes:

IMAS

https://downloadmirror.intel.com/29337/eng/Intel_Memory_And_Storage_Tool_User%20Guide-Public-342245-001US.pdf

<https://downloadcenter.intel.com/download/29337/Intel-Memory-and-Storage-Tool-CLI-Command-Line-Interface- Commands>

intelmas show -o json -intelssd intelmas show -intelssd 1 -identify intelmas dump -destination telemetry_data.bin -intelssd 1 -telemetrylog
intelmas dump -destination telemetry_data.bin -intelssd 1 -eventlog

static createDir(dirName)

function for changing cwd to the directory specified in dirName

Args:

dirName: String for path to destination directory

Returns:

static findExecutable(executable='', path=None)

function for finding 'executable' in the directories listed in 'path'.

Args:

executable: the name of the executable file path: A string listing directories separated by 'os.pathsep'; defaults to os.environ['PATH'].

Returns:

the complete filename or None if not found.

genericParseTelemetry(fileList=None)

function for parsing a list of binary files

Args:

fileList: List of binary files' names or paths

Returns:

A list of names for the folders where the parsed binary files are stored

getDebug()

function for reading the debug flag stored in the generateTSBinaries attributes

Returns:

Boolean flag to activate debug statements

getIterations()

function for reading the iterations stored in the generateTSBinaries attributes

Returns:

Integer number of data points to be considered in the time series

getOutpath()

function for reading the outpath stored in the generateTSBinaries attributes

Returns:

String for the path of the output directory where the binaries will be stored

hybridGetBinary(*time=None, device=None, identifier='Tv2HiTAC'*)

function to generate the binary files using the nvme-cli commands

Args:

time: Integer time (in seconds) for which data will be collected

device: String for name of device interface to get data from

identifier: String for the name of the data set that corresponds to the telemetry pull to be executed

Returns:

Two lists: one for files and another one for folders generated

hybridGetTelemetry(*time=None, start=None, device=None, identifier='Tv2HiTAC'*)

function for getting telemetry binary using the nvme-cli interface

Args:

time: Integer time (in seconds) for which data will be collected

start: Time at which the operation is supposed to start recording

values device: String for name of device interface to get data from

identifier: String for the name of the data set that corresponds to the telemetry pull to be executed

Returns:

A list of names for the telemetry bin files generated

`intelmasGetBinary(time=None, driveNum=None,
identifier='Tv2HiTAC')`

function to generate the binary files using the intelmas commands

Args:

time: Integer time (in seconds) for which data will be collected
driveNum: Integer value for the number of the drive from which telemetry data will be pulled
identifier: String for the name of the data set that corresponds to the telemetry pull to be executed

Returns:

Two lists: one for files and another one for folders generated

`intelmasGetTelemetry(time=None, start=None, driveNum=None,
identifier='Tv2HiTAC')`

Function for getting telemetry binary using intelmas commands

Args:

time: Integer time (in seconds) for which data will be collected
start: Time at which the operation is supposed to start recording
values
driveNum: Integer value for the number of the drive from which telemetry data will be pulled
identifier: String for the name of the data set that corresponds to the telemetry pull to be executed

Returns:

A list of names for the telemetry bin files generated

`internalGetBinary(time=None, driveNum=None,
identifier='Tv2HiTAC')`

function to generate the binary files using the internal TWIDL commands

Args:

time: Integer time (in seconds) for which data will be collected
driveNum: Integer value for the number of the drive from which

telemetry data will be pulled identifier: String for the name of the data set that corresponds to the telemetry pull to be executed

Returns:

Two lists: one for files and another one for folders generated

`internalGetTelemetry(time=None, start=None, driveNum=None, identifier='Tv2HiTAC')`

Function for getting telemetry binary using the TWIDL interface

Args:

time: Integer time (in seconds) for which data will be collected

start: Time at which the operation is supposed to start recording

values driveNum: Integer value for the number of the drive from which telemetry data will be pulled identifier: String for the name of the data set that corresponds to the telemetry pull to be executed

Returns:

A list of names for the telemetry bin files generated

`parseBinaryOnly(inputFolder=None)`

function for parsing all binary files contained in a given directory

Args:

inputFolder: String of the relative path to the directory containing all the binary files to be parsed

Returns:

Two lists: one for files parsed and another one for folders generated

`setDebug(debug)`

function for setting the debug flag stored in the generateTSBinaries attributes

Returns:

`setIterations(iterations)`

function for setting the iterations stored in the `generateTSBinaries` attributes

Returns:

`setOutpath(outpath)`

function for setting the outpath stored in the `generateTSBinaries` attributes

Returns:

`src.software.TSV.generateTSBinaries.main()`

main function to be called when the script is directly executed from the command line

`visualizeTS.py`

This module contains the basic functions for plotting time series generated from a configuration file. Multiple transformations to the data can be applied using the enhance functions before visualization. To run this script you must use a GUI, as pyplot requires the environment variable `$DISPLAY` to be set to generate the PDF files. The execution of this module assumes that the configuration file only contains data for a single core; data for multiple cores in a single config file will result in distorted graphs

Args:

--outfile: String for the name of the output file (without the suffix) --targetObject: String for the object name to be processed --targetFields: Comma-separated strings for the names of the object's flatten fields to be processed --inputFile: String of the name for the configuration file containing the data for the time-series --combine: Boolean flag to indicate whether or not all fields should be combined in a single plot --subSeqLen: Integer value for the length of the sliding window to be used for generating the matrix profile --transformTS: Boolean flag to

generate the matrix profile for the given time series --debug: Boolean flag to activate debug statements

Example:

Default usage:

```
$ python visualizeTS.py
```

Specific usage:

```
$ python visualizeTS.py --outfile test1 --targetObject
ThermalSensor --targetFields logtimer,secondstimer
--inputFile time-series/time-series.ini --debug True
```

`src.software.TSV.visualizeTS.main()`

main function to be called when the script is directly executed from the command line

`class src.software.TSV.visualizeTS.visualizeTS(debug=False)`

Bases: `object`

`check_stationarity(timeseriesCandidate)`

Augmented Dickey-Fuller (ADF) test can be used to test the null hypothesis that the series is non-stationary. The ADF test helps to understand whether a change in Y is a linear trend or not. If there is a linear trend but the lagged value cannot explain the change in Y over time, then our data will be deemed non-stationary. The value of test statistics is less than 5% critical value and p-value is also less than 0.05 so we can reject the null hypothesis and Alternate Hypothesis that time series is Stationary seems to be true. When there is nothing unusual about the time plot and there appears to be no need to do any data adjustments. There is no evidence of changing variance also so we will not do a Box-Cox transformation.

Args:

`timeseriesCandidate: Time series`

Returns: Determination of whether the data is stationary or not?

`generateMP(dataDict, obj=None, fields=None, subSeqLen=20,
visualizeAllObj=True, visualizeAllFields=True)`

function for generating a matrix profile for multiple time series contained in *dataDict*

Args:

dataDict: Dictionary containing all the time-series data
obj: List of objects for which to generate a file containing all the desired plots
fields: List of fields for which to generate a plot inside the output file
subSeqLen: Integer value for the length of the sliding window to be used to generate the matrix profile
visualizeAllObj: Boolean flag indicating that all objects in the configuration file should be considered
visualizeAllFields: Boolean flag indicating that all fields for an object should be plotted

Returns:

A dictionary containing all the data for the matrix profiles associated with the given fields of the specified objects

`generatePDFFile(dataDict, objectID, outfile, fields, visualizeAllFields,
combine, start=0, end=100)`

function for generating a PDF file that contains the time-series plot for the data contained in *dataDict*

Args:

dataDict: Dictionary containing all the time-series data
objectID: String for the name of the object to be processed (ex. uid-6)
outfile: String of the prefix for the output file
fields: List of fields for which to generate a plot inside the output file
visualizeAllFields: Boolean flag indicating that all fields for an object should be plotted
combine: Boolean flag indicating whether or not to combine all fields in a single graph
start: Integer value for the start percentage of the lists
end: Integer value for the end percentage of the lists

Returns:

`generatePDFPlots(subdict, ax, column, combine, pdf, start=0, end=100)`

function for plotting the time-series and saving the plots in the specified pdf file

Args:

ax: matplotlib axis object where the data will be plotted
subdict: Dictionary containing all the data for an object
column: String for the name of the field to be processed
combine: Boolean flag indicating whether or not to combine all fields in a single graph
pdf: File descriptor for the pdf file where the plots will be stored
start: Integer value for the start percentage of the lists
end: Integer value for the end percentage of the lists

Returns:

`generateTSVisualizationCMD(subdict, objectID, fields, visualizeAllFields, combine, pdf, start=0, end=100)`

function to generate the time-series plots using command-line parameters

Args:

objectID: string for the object id to be used in the title
subdict: Dictionary containing all the data for an object
fields: List of fields for which to generate a plot inside the output file
visualizeAllFields: Boolean flag indicating that all fields for an object should be plotted
combine: Boolean flag indicating aggregate all fields in a single graph
pdf: File descriptor for the pdf file where the plots will be stored
start: Integer value for the start percentage of the lists
end: Integer value for the end percentage of the lists

Returns:

`generateTSVisualizationGUI(name, subdict, mainFields, secondaryFields, start=0, end=100)`

function for generating the time-series plots using GUI parameters and options

Args:

name: String for the name of the graph subdict: Dictionary containing all the data for an object mainFields: List of strings for the fields to be graphed in the primary axis secondaryFields: List of strings for the fields to be graphed in the secondary axis start: Integer value for the start percentage of the lists end: Integer value for the end percentage of the lists

Returns:

getDebug()

function for reading the debug flag stored in the visualizeTS attributes

Returns:

Boolean flag to activate debug statements

populateMPStruct(*MP, subdict, object_t, subSeqLen, fields, visualizeAllFields*)

function for generating the dictionary with the matrix profile values for the time-series

Args:

MP: Dictionary where all the data lists will be stored subdict: Dictionary where the unprocessed data lists are contained object_t: String for the name of the object for which we are extracting the matrix profiles (ex. uid-6) subSeqLen: Integer for the window size to be used in the matrix profile generation fields: List of strings for the fields to be processed visualizeAllFields: Boolean flags to process all fields

Returns:

setDebug(*debug*)

function for setting the debug flag stored in the visualizeTS attributes

Args:

debug: Boolean flag to activate debug statements

Returns:

`visualizeTSAPI(obj=None, fields=None, input_t='time-series.ini', out='telemetryDefault', combine=False, subSeqLen=20, transformTS=False, visualizeAllObj=False, visualizeAllFields=False, raw_uid=False, inParallel=False, requiredList=None, timeOut=180)`

API to replace standard command line call (the visualizeTS class has to instantiated before calling this method)

Args:

inParallel: Flag to process all objects in parallel processes.

raw_uid: Boolean flag to indicate whether the uid contains the prefix 'uid-' requiredList: List of strings for the names of objects

to be processed if the useRequiredList flag is set.

If None, the default list will be used. Indicates whether the objects to be processed should be limited to the ones contained in the requiredList.

obj: List of object UIDs for which to generate a file containing all the desired plots fields: List of fields for which to generate a plot inside the output file input_t: String of the name for the configuration file containing the data for the time-series out: String of the prefix for the output file combine: Boolean flag indicating whether or not to combine all fields in a single graph subSeqLen: Integer value for the length of the sliding window to be used to generate the matrix profile transformTS: Boolean flag indicating that the matrix profile for the time series will be generated visualizeAllObj: Boolean flag indicating that all objects in the configuration file should be considered

`visualizeAllFields`: Boolean flag indicating that all fields for an object should be plotted
`timeOut`: Execution timeout

Returns:

`class visualizeUtility`

Bases: `object`

`static generateTruncatedLists(subdict, field, start, end)`

function for generating the data list of the corresponding percentage of values

Args:

`subdict`: Dictionary containing the data and values field:
String for the name of the field to be accessed in the dictionary
`start`: Integer value for the start percentage of the lists
`end`: Integer value for the end percentage of the lists

Returns:

`dataList`: list containing the data values for the corresponding percentages

`writeTSVisualizationToPDF(dataDict, obj=None, outfile='telemetryDefault', fields=None, combine=False, subSeqLen=20, transformTS=False, visualizeAllObj=False, visualizeAllFields=False, raw_uid=False, inParallel=False, timeOut=180)`

Function for generating a basic line plot for the time series data. It is assumed that the data is in order and its index represents the relative time of collection

Args:

`inParallel`: Flag to process all objects in parallel processes.
`raw_uid`: Boolean flag to indicate whether the uid contains the prefix 'uid-'
`dataDict`: Dictionary containing all the time-series data
`obj`: List of objects for which to generate a file containing all the desired plots
`outfile`: String of the prefix for the output

file fields: List of fields for which to generate a plot inside the output file
combine: Boolean flag indicating whether or not to combine all fields in a single graph
subSeqLen: Integer value for the length of the sliding window to be used to generate the matrix profile
transformTS: Boolean flag indicating that the matrix profile for the time series will be generated
visualizeAllObj: Boolean flag indicating that all objects in the configuration file should be considered
visualizeAllFields: Boolean flag indicating that all fields for an object should be plotted
timeOut: Time before aborting computation.

Returns:

class src.software.mp.order.Order

Bases: **object**

An object that defines the order in which the distance profiles are calculated for a given Matrix Profile

next()

class src.software.mp.order.linearOrder(*m*)

Bases: [src.software.mp.order.Order](#)

An object that defines a linear (iterative) order in which the distance profiles are calculated for a given Matrix Profile

next()

Advances the Order object to the next index

class src.software.mp.order.randomOrder(*m*, *random_state=None*)

Bases: [src.software.mp.order.Order](#)

An object that defines a random order in which the distance profiles are calculated for a given Matrix Profile

next()

Advances the Order object to the next index

src.software.dAMP.gatherMeta.API(*options=None*)

API for the default application in the graphical interface. Args:

options: Commandline inputs.

Returns:

class

src.software.dAMP.gatherMeta.GatherMeta(*binaryPath='data/inputSeries/'*,
fwDir='..//Auto-Parse/decode/ADP_UV', *dataCntrlStructPath='Auto-*
Parse/datacontrol/structures.csv', *configFileName='data/workload-healthy-*
ADP.ini', *resultsFolder='data/output'*,
nlogFolder='software/parse/nlogParser', *useCSV=False*, *debug=False*)

Bases: **object**

ARMAFigures = *None*

CFG = *None*

DFA = *None*

DFG = *None*

DataLakeMeta = *None*

MLMinedJira = *None*

MLProfiles = *None*

RNNAccuracy = *None*

RNNFigures = *None*

assistedFigures = *None*

binaryPath = *None*

configData = *None*

configFileName = *None*

constructSystemInfo()

dataCntrlStructPath = *None*

dataDict = *None*

dataDictDimension = *None*

debug = *False*

deviceConfiguration = *None*

deviceSignature = *None*

extractTelemetryHeader()

function for extracting the telemetry header

Returns:

telemetryHeader = dictionary for telemetry data object
containing the header information

firmwareDFA = *None*

fwDir = *None*

getARMAFigures()

getApplications()

Applications [List] |-> Application (uID, Name, Version, Functions)

Returns:

getAssistedFigures()

Generated figures based on pertinent information highlighted in handbook. Returns:

getClasstoDictionary()

getDataLakeMeta()

Data Lake: ([Identifiers: “123456789”], [URL: <https://datalake.intel.com/browse/key/>], Major: [1 to M], Minor: [0 to K]

getDeviceConfiguration(*sigMode='latest'*, *index=0*)

Solid State Device configuration Returns: (Meta of SSD, Factory Configuration of SSD)

getDeviceSignature(*sigMode='index'*, *index=0*)

Return the telemetry header state of most recent Node or Timeseries. *sigMode*: Mode of accessing data either by a given index or the entire series. Returns:

getFirmwareDFA(*gitURL=None*, *changeset=None*)

Control, Data Flow state and transitions maps from firmware. Args:

gitURL: Source Code Repository *changeset*: Changeset associated with the code change.

Returns:

getFlattenDictionary(*dd*, *separator='_'*, *prefix=''*)

getInfo()

getJIRA()

JIRA: (Identifiers: [“NSGSE-12345”], [URL: <https://nsg-jira.intel.com/browse/>], Major: [1 to M], Minor: [0 to K])

getMLClustering()

getMLMinedJira()

getMLProfiles()

Data labeled based on the state from models. Returns:

getNlogInfo()

getRNNFigures()
getSuperDictionary()
getSystemInfo()

getTelemetryMetaStats()

Labeling of data for Machine Programming Learning systems
Returns:

Telemetry data machine learning information. [UID List]

| -> [Time indexs] |
| -> (dataSets, width, length)
| -> Meta Sets width, length

Telemetry: Telemetry information in respect the current payload

Internal: Telemetry version is the internal tracking across all products.

Major: [1 to M] Minor: [0 to K]

Protocol Interface: = ["NVMe", "SATA-ACS"] - The specific protocol used. Type: ["Host", "Controller"] - Host or controller-initiated telemetry asynchronous command.
Snapshots: Telemetry binary payloads names and path pairs.

Filenames: "PHAB8506001G3P8AGN_sample.bin" - File name on the host system. Paths:

"C:/Source/SSDDev/NAND/gen3/tools/telemetry/sample" - Path on the host system to binary.

PayloadValidity: ["Verified", "Unknown", "Corrupted"] creationTime: [year, month, day[, hour[, minute[, second[, microsecond[, tzinfo]]]]]] -

Represents the time at which the data object was imported to the tool.

`getTimeSeries()`

function for obtaining the dictionary of the time series data Returns:

`getUIDDescriptions(uidsFound: Optional[list] = None)`

Args:

`uidsFound`: list of uids with the format 'uid-#'.

Returns: dictionary that describes uids (key) with its global name and definition (value).

`static getUIDValue(stringInput=")`

`nlogFolder = None`

`nlogTable = None`

`objectsOfInterest = ['uid-41', 'uid-44', 'uid-45', 'uid-46', 'uid-47', 'uid-48', 'uid-49', 'uid-58', 'uid-139', 'uid-145', 'uid-153', 'uid-172', 'uid-181', 'uid-182', 'uid-185', 'uid-191', 'uid-196', 'uid-198', 'uid-200', 'uid-201', 'uid-205', 'uid-216', 'uid-219', 'uid-229', 'uid-235', 'uid-236', 'uid-237', 'uid-300', 'uid-301', 'uid-307', 'uid-382', 'uid-1000140', 'uid-1000170']`

`org`

`resultsFolder = None`

`setARMAFigures(inputFile=None, matrixProfile=None,
subSeqLen=None, targetObject=None, targetField=None,
debug=None)`

`setApplications()`

Applications [List] |-> Application (uID, Name, Version, Functions)
Returns:

`setAssistedFigures(outfile=None, numCores=None, driveType='ADP')`

Generated figures based on pertinent information highlighted in handbook. Returns:

```
setAssistedFiguresParallel(outfile=None, numCores=None, driveType='ADP', max_workers=None, timeOut=180, inOrder=True, runSequential=True, debug=False)
```

Generated figures based on pertinent information highlighted in handbook. Returns:

```
setDataLakeMeta(axonID, axonURL='https://axon.intel.com', MetaDataFile=None, UploadedFile=None, DownloadedFile=None)
```

Data Lake: ([Identifiers: “123456789”], [URL: <https://datalake.intel.com/browse/key/>], Major: [1 to M], Minor: [0 to K])

```
setDeviceConfiguration()
```

Solid State Device configuration Returns: (Meta of SSD, Factory Configuration of SSD)

```
setDeviceSignature()
```

Return the telemetry header state of most recent Node or time series. sigMode: Mode of accessing data either by a given index or the entire series. Returns:

```
setFirmwareDFA(gitURL=None, changeset=None)
```

Control, Data Flow state and transitions maps from firmware. Args:

gitURL: Source Code Repository changeset: Changeset associated with the code change.

Returns:

```
setJIRA()
```

JIRA: (Identifiers: [“NSGSE-12345”], [URL: <https://nsg-jira.intel.com/browse/>], Major: [1 to M], Minor: [0 to K])

```
setMLClustering()
```

setMLMinedJira(*nearNeighborTables=None*,
jiraClusteringPDFList=None, *tableTitles=None*)

setMLProfiles(*subSeqLen=None*)

 Data labeled based on the state from models. Returns:

setNlogInfo(*nlogTable=None*)

setRNNFigures(*configFilePath=None*, *inputWidth=None*,
labelWidth=None, *shift=None*, *batchSize=None*, *hiddenLayers=None*,
targetObject=None, *targetFields=None*, *plotColumn=None*,
maxEpochs=None, *matrixProfileFlag=None*, *subSeqLen=None*,
embeddedEncoding=None, *categoricalEncoding=None*, *debug=None*)

setTelemetryMetaStats()

setTimeSeries(*extractBinaries=True*, *debug=False*)

 function for generating the dictionary of the time series data

 Returns:

systemInfo = *None*

telemetryHeaderDict = *None*

telemetryMetaStats = *None*

timeSeriesSignatures = *None*

toString()

uidsFound = *None*

updateInitialize(*binaryPath=None*, *fwDir=None*,
dataCntrlStructPath=None, *nlogFolder=None*, *configFileName=None*,
resultsFolder=None, *extractBinaries=False*, *useCSV=False*,
debug=False)

 function for updating the initial values of the analysisReport

Args:

binaryPath: relative path to the folder where the telemetry binaries are stored fwDir: relative path to the folder where the firmware parsers are stored dataCntrlStructPath: relative path to the file where the data control structures are stored nlogFolder: Folder containing negative logs and tokenizers configFileName: String representation of the name of the .ini file where the time series is stored or will be stored resultsFolder: relative path to folder where the results of intermediate operations will be stored extractBinaries: boolean flag that indicates the choice using binary files in generate a time series automatically useCSV: Boolean flag to indicate that the file to be used is a .csv file instead of the .ini file debug: boolean flag for verbose printing

updateReportMeta(uidsFound=None, deviceConfiguration=None, dataDict=None, dataDictDimension=None, DFA=None, CFG=None, DFG=None, MLPProfiles=None, jiraNeighbors=None, jiraFigures=None, tableTitles=None, nlogTable=None, assistedFigures=None, timeSeriesSignatures=None)

version

src.software.dAMP.gatherMeta.main()

src.software.dAMP.gatherMeta.testSuite(*debug: bool = False*)

Function Type: Method Run all test for the class.

src.software.dAMP.reportGenerator.API(*options=None*)

API for the default application in the graphical interface. Args:

options: Commandline inputs.

Returns:

class

src.software.dAMP.reportGenerator.RegressionUnitTestsCases(*outDir: Optional[str] = None, referenceDir: Optional[str] = None, debug: bool = True*)

Bases: `object`

Self regression testing class for report generator. The following is a test set to ensure minimum functionality.

`debug = True`

`filename = None`

`getTestFunctionList()`

`homeDir = None`

`reportGenObj = None`

`run()`

Function Type: Method

Run the test, collecting the result into the test result object passed as

`result`. If `result` is omitted or `None`, a temporary result object is created (by calling the `defaultTestResult()` method) and used. The result object is not returned to [`run\(\)`](#)'s caller.

The same effect may be had by simply calling the `TestCase` instance.

`setUp()`

Function Type: Helper # Method called to prepare the test fixture. This is called immediately # before calling the test method; any exception raised by this method will # be considered an error rather than a test failure. The default # implementation does nothing.

`status = 'Pass'`

`suite()`

Method called to prepare the test fixture. This is called immediately before calling the test method; any exception raised by this method will be considered an error rather than a test failure. The

default implementation does nothing.

`tearDown()`

Function Type: Helper

Method called immediately after the test method has been called and the

result recorded. This is called even if the test method raised an exception, so the implementation in subclasses may need to be particularly careful about checking internal state. Any exception raised by this method will be considered an error rather than a test failure. This method will only be called if the [`setUp\(\)`](#) succeeds, regardless of the outcome of the test method. The default implementation does nothing.

`test_addMathMatrix()`

Template mathematics matrix testing. Returns: Mathematics object.

`test_addMetaTable()`

Template meta table testing. Returns: None.

`test_addPDFImages()`

Template PDF addition testing. Returns: None or error string.

`test_createDocument()`

Create a document and compare it to reference. Create a custom diff tool for comparing .tex files. Returns: Status of test.

`test_updateCover()`

Template cover testing. Returns: None.

`test_updateCustomerInformation()`

Template customer testing. Returns: None.

*class src.software.dAMP.reportGenerator.ReportGenerator(`outPath=None`,
`fileName=None`, `logoImage=None`, `debug: bool = True`)*

Bases: **object**

Report generation in LaTeX to PDF.

`addCustomCommandText(candidateString: str)`

`addCustomlineText(candidateString: str)`

`static addEnumerateList(enumStrList: Optional[list] = None)`

`static addImagesRows(inFigureList: Optional[list] = None, rowWidth: int = 1, image_options: Optional[list] = None)`

Add PDF images to report in a simple n by m column format. Args:

inFigureList: Figure string list of paths to add. rowWidth: width of row of images image_options: string list – Specifies the options for the image (ie. height, width)

Returns: Figure table object and total rows in existence.

`static addMathMatrix(arrayMatrix: Optional[list] = None, arrayMatrixLabel: Optional[str] = None)`

Add a matrix table in mathematical form. Args:

arrayMatrix: Array to add into document. arrayMatrixLabel: Array label to be on the left hand side of the equal operator.

Returns: mathematics object for report.

`addMetaTable(tableName: Optional[str] = None, tableLayout: Optional[str] = None, tableHeader: Optional[List[str]] = None, tableMeta: Optional[list] = None, alternateColoring: int = 2, color: str = 'lightgray', mapper=<function bold>, row_height: float = 1.5)`

Meta data table constructor. Args:

tableName: Name of table. tableLayout: Format of table.

tableHeader: Header of table tableMeta: Table meta data.

alternateColoring: Coloring for human friendliness. color:
Color setting. mapper: mapper for rows. row_height: row
customization height.

Returns: Tablename and table object pair.

*static addPDFImages(*inFigureList*: Optional[List[str]] = None)*

Add PDF images to report in a simple two column format. Args:

inFigureList: Figure string list of paths to add.

Returns: Figure table object and total rows in existence.

*static addPDFImagesRows(*inFigureList*: Optional[list] = None,
rowWidth: int = 1)*

Add PDF images to report in a simple n by m column format. Args:

inFigureList: Figure string list of paths to add. *rowWidth*:

RReturns: Figure table object and total rows in existence.

authorName = *None*

codeBranchPage = *None*

collectionTypes = *None*

createDocument()

Documentation creation module to streamline all components.

Returns:Void

*createDocumentRAAD(systemInfo=**None*)

Documentation creation module to streamline all components.

Returns:Void

*createDocumentRAAD_Backup(systemInfo=**None*)

Documentation creation module to streamline all components.

Returns:Void

`static filePath(requestPath)`

Generate file path with existance okay. Args:

requestPath: path to request creation.

Returns: Void

`customerName = None`

`customerPage = None`

`debug = True`

`dict2BulletList(elements=None, level: int = 1, indent_size: int = 4,
spaceOrTab: bool = True, addNewLine: bool = True, retSpaceValue: bool = True)`

Recursive method to traverse a dictionary, list, set, basic type to create a tab based index and individual item list. Args:

elements: traversing dictionary
level: level variable for recursive indentition
indent_size: size of indent in spaces
spaceOrTab: Flag to use spaces or tabs. *addNewLine*: Add new line to delimited item. *retSpaceValue*: Flag to return space values.

Returns: Formatted tupple with tab/spaces indicated.

`extractFiles(inTokens, fileType: str = '.pdf')`

Function to traverse a dictionary and capture the key and values related to PDF or a given file type. Args:

inTokens: dictionary, list, set, or string
fileType: File extension to detect.

Returns: dictionary of found items with file type.

`fileName = None`

`firmwareName = None`

`firstPage = None`

`static generateString(characterSet=None, fileStringNameSize=None)`

Generate random string set when not set by user. Args:

`characterSet`: Character set to be used. `fileStringNameSize`: File string name size maxima.

Returns: Token generated.

`geometry_options = None`

`static getLiteral(varObj)`

`static getTempFileName(genFile=True)`

Generate file name. Args:

`genFile`: proceed flag with generation.

Returns: file name token string.

`getTempPathAndFileName(extensionName=None, genPath=True, genFile=True)`

Generate path and file name. Args:

`extensionName`: Extension name desired. `genPath`: proceed flag with generation. `genFile`: proceed flag with generation.

Returns: token generation of path and file name token string.

`static getTempPathName(genPath=True)`

`static hyperlink(url: Optional[str] = None, text: Optional[str] = None)`

`logoImage = None`

`modelName = None`

`outPath = None`

`pendingDocument = None`

`productName = None`

`productSerial = None`

`productStatus = None`

`reportVersion = None`

`savePath = None`

`setDebug(status: bool = True)`

Set debug status of class. Args:

status: flag to enable and disable debug status.

Returns: Void

`setGeometry(head=None, margin=None, bottom=None, includeheadfoot=None)`

Set the page layout to target for PDF/printing. Args:

head: Header of file setup. margin: Page margin. bottom: Page bottom margin. includeheadfoot: Add header/footer.

Returns: Void

`setInitialize(outPath=None, fileName=None, logoImage=None)`

Setup of the class configuration basis. Args:

outPath: Path to create/save. fileName: Desired file name. logoImage: Desired logo to be added to the report.

Returns: Void

`specificationName = None`

`specificationVersion = None`

`supportedProductList = None`
`teamName = None`
`timeToken = None`
`titleInfo = None`
`updateCover(titleInfo=None, teamName=None, authorName=None, specificationName=None, specificationVersion=None, supportedProductList=None, reportVersion=None)`

Cover template for the report. Args:

`titleInfo`: Title to be presented. `teamName`: Team name used in the gather process. `authorName`: Author of report generation. `specificationName`: System specification and name. `specificationVersion`: Version number of specification. `supportedProductList`: Product list within report or supported. `reportVersion`: Report module version.

Returns: Void

`updateCustomerInformation(customerName=None, userName=None, productName=None, modelName=None, productStatus=None, productSerial=None, firmwareName=None, collectionTypes=None)`

Customer template information setup. Args:

`customerName`: Customer name `userName`: Interface user name. `productName`: Product name used. `modelName`: Product model used. `productStatus`: Product status. `productSerial`: Identifier of the product. `firmwareName`: Firmware used in extraction. `collectionTypes`: Collection set to be used for analysis, regression, and forecasting.

Returns: Void.

`updateTimeStamp(inTime=None)`

Time stamp update or set for document. Args:

inTime: date time desired to be set.

Returns: Void

user_name = *None*

src.software.dAMP.reportGenerator.main()

src.software.dAMP.reportGenerator.testSuite(*debug: bool = False*)

Function Type: Method Run all test for the class.

class src.software.dAMP.iSOM.iSOM(*debug: bool*)

Bases: **object**

System self-organizing map (SOM) to investigate the structure of a set of data.

With uncertainty in meta, SOM is used to discover and indicate classes within data through dimensionality reduction. To proceed create a U-Matrix (NxM), the cells contain a value to indicate data items that are similar to each other with extremes in the range indicating boundaries; where N indicates data samples in rows and M indicates measured features. These gradients are used to infer graph partitions in space of similar classes. A cell vector closest to a specified data item is called the best matching unit (BMU).

Algorithm (meta-heuristic)

create n x m, map with random node vector values while s < StepsMax times compute what a "close" node means, based on s compute a learn rate, based on s pick a random data item determine the map node closest to data item (BMU) for-each node close to the BMU

adjust node vector values towards data item

Note: Instead of using an n-by-m rectangular grid, you can use a layout where each cell in the SOM is a hexagon.

You can use a toroidal geometry where edges of the SOM connect. You can use three dimensions instead of two. There are many ways to define a close neighborhood for nodes.

Reference: Dr. James McCaffrey Microsoft Research -
jamccaff@microsoft.com

Cols = *None*

Dim = *None*

LearnMax = *None*

RangeMax = *None*

Rows = *None*

StepsMax = *None*

data_x_shadow = *None*

data_y_shadow = *None*

debug = *False*

static distance_euclidean(v1, v2)

Euclidean distance between two vectors. v1 = (2, 1, 4) and v2 = (5, 1, 8) then euc_dist() = $\sqrt{(5 - 2)^2 + (1 - 1)^2 + (8 - 4)^2}$ = $\sqrt{25} = 5.0$ Args:

v1: v2:

Returns:

static distance_manhattan(r1, c1, r2, c2)

Manhattan distance between two cells with coordinates (r1, c1) and (r2, c2) (2, 5) and (6, 8) is $4 + 3 = 8$ Args:

r1: c1: r2: c2:

Returns:

static feature_intersection(lst: list, n)

Common value from a list of integer values. [0, 2, 2, 1, 0, 1, 1, 2, 1]
then 1 Args:

lst: n:

Returns:

static nearest_neighbor(data, t, maping, m_rows, m_cols)

static normalize_data(inArray)

Divide an array by its norm to normalize the array. Args:

inArray:

Returns:

runPhases()

Common reports Retrieves data for common reports

class src.software.JIRA.classCommonReports.common_reports

Bases: **object**

Generates all of the html output # <p>

allDevCommits(aIssuesList, aSupportFile)

Separates the list of issues by developer (dev Owner, Val Owner (if in FPV)) # This represents only things they are currently working (have their name not) not commits/stretches # The priority is not filtered in the returns, so all priority levels are in the list # Note: Uses the team list of issues, so that we don't have to filter by team # # @param aIssuesList Filename to open # # @Returns Returns a dictionary of list developers and list of the issues that they are working on

commitsByDev(aIssuesList, aSupportFile)

```
# # Separates the list of issues by developer (dev Owner, Val Owner  
# (if in FPV)) # This represents only things they are currently working  
# (have their name not) not commits/stretches # The priority is not  
# filtered in the returns, so all priority levels are in the list # Note:  
# Uses the team list of issues, so that we don't have to filter by team #  
# @param aIssuesList Filename to open # # @Returns Returns a  
# dictionary of list developers and list of the issues that they are  
# working on #
```

htmlOutput = *None*

htmlOutputFileName = ''

sightingsByDev(*aIssuesList*, *aSupportFile*)

```
# # Separates the list of issues by developer (dev Owner, Val Owner  
# (if in FPV)) # This represents only things they are currently working  
# (have their name not) not commits/stretches # The priority is not  
# filtered in the returns, so all priority levels are in the list # Note:  
# Uses the team list of issues, so that we don't have to filter by team #  
# @param aIssuesList Filename to open # # @Returns Returns a  
# dictionary of list developers and list of the issues that they are  
# working on #
```

sightingsBCH(*aIssuesList*)

```
# # Gathers a list of all BCH sightings in dev, val or FPV. # Note:  
# Uses the team list of issues, so that we don't have to filter by team #  
# @param aIssuesList Filename to open # # @Returns Returns a list  
# of all BCH issues in dev, val, or FPV #
```

sightingsSummaryByProgram(*aIssuesList*)

```
# # Gathers count of issues in each priority by program # Note: Uses  
# the team list of issues, so that we don't have to filter by team # #  
# @param aIssuesList Filename to open # # @Returns Returns a  
# dictionary with keys of the program name and list of number of  
# priority of sightings #
```

urlRoot = '<http://nsg-jira.intel.com>'

Scans the snapshot for things that happened this week and produces a report
jira_issues_lists class converts the fields read from the JSON file into a
groovy list for easier consumption by calling routines.

class src.software.JIRA.classJiraIssues.jira_issue

Bases: **object**

```
# jira_issues class parses a single issue from JSON into variables # <p>
# Note: These variables are used by the jira_issues_list class to # make
the lists of issues # # @version 1.0 #
```

parseJson(jsonString)

```
# parseJson ## Parses a single Json string and extracts the parts of
the issue we are interested in. # populates this classes variables with
those part of interest # <p> # # @param jsonString the string to
parse
```

*class src.software.JIRA.analysisGuide.AnalysisGuide(username=None,
password=None, debug=False,
defaultPasswordFile='../../raadProfile/credentials.conf')*

Bases: **object**

class FirmwareAssert(assertCodeValue, debug=False)

Bases: **object**

getDebugInfo()

class FirmwareDataForAnalysis(uid, name)

Bases: **object**

```
static create_temporary_fileSplit_copy(path='.',
fileCopyName='real.dump', tempFileName='temp_file_name.dump')
```

Args:

path: fileCopyName: tempFileName:

Returns:

*static createTemporaryFileCopy(fileCopyName=None,
tempFileName=None)*

Args:

fileCopyName: tempFileName:

Returns:

getAllLinks(startURL: Optional[str] = None, findTag: Optional[str] = None)

Args:

startURL: findTag:

Returns:

getHandbookInfo()

getINIFileClass(loadFile='../../raadProfile/debugHandbookCache.ini')

Args:

loadFile:

Returns:

getINIFileDictionary(loadFile='../../raadProfile/debugHandbookCache.ini')

returns a dictionary with keys of the form <section>.<option> and the corresponding values

*get_allMeta(username=None, password=None,
outFileData='queryTesting.txt', debug=True)*

F Args:

username: password: outFileData: debug:

Returns:

handbookInfo = *None*

loadAll(*loadFile*=*'../../../../raadProfile/debugHandbookCache.ini'*)

Args:

loadFile:

Returns:

loadCache(*loadFile*=*'../../../../raadProfile/debugHandbookCache.ini'*)

Args:

loadFile:

Returns:

loadCacheAndSearch(*loadFile*=*'../../../../raadProfile/debugHandbookCache.ini'*, *signature*=*'ASSERT_DE003'*, *similarityScore*=*0.95*)

Args:

loadFile: *signature*: *similarityScore*:

Returns:

loadLive(*saveFile*=*'../../../../raadProfile/debugHandbookCache.ini'*)

Args:

saveFile:

Returns:

loadedINI = *None*

pageLogin(*urlPageBase*, *urlPageExtLogin*, *login*, *password*)

Args:

urlPageBase: *urlPageExtLogin*: *login*: *password*:

Returns:

performUserInputAPI(*selectedMode*=*2*,
defaultFile=*'../../../../raadProfile/credentials.conf'*,

*default_username='PresidentSkroob@spaceballs.one',
default_password='12345', outputFile='../../data/queryTesting.txt')*

Args:

selectedMode: defaultFile: default_username: default_password:
outputFile:

Returns:

*pythonAPI(selectedMode=2,
defaultFile='.raadProfile/credentials.conf',
default_username='PresidentSkroob@spaceballs.one',
default_password='12345',
outputFile='.raadProfile/debugHandbookCache.ini',
searchString='ASSERT_DE003', similarityScoreThreshold=0.95)*

Args:

selectedMode: defaultFile: default_username: default_password:
outputFile: searchString: similarityScoreThreshold:

Returns:

queryWebpage(assertCode=None, outFileData='queryResult.txt')

Args:

assertCode: outFileData:

Returns:

searchForSignature(signature='ASSERT_DE003', similarityScore=1.0)

Searches for specific signature Args:

signature: similarityScore:

Returns:

*searchForSignatureFromCode(signature='ASSERT_DE003',
similarityScore=1.0)*

Signature search and match threshold Args:

signature: similarityScore:

Returns:

setUser(*username=None, password=None*)

Args:

 username: password:

Returns:

setUsernamePassword(*username=None, password=None*)

verifyAccess()

Returns:

writeINIFile(*saveFile='../../../../raadProfile/debugHandbookCache.ini'*)

Args:

 saveFile:

Returns:

class src.software.JIRA.analysisGuide.HandbookMeta(*code=None, analysisUIDs=None, knownCauses=None, url=None*)

Bases: **object**

set_analysisUIDs(*extractedValue=None*)

Args:

 extractedValue:

Returns:

set_code(*extractedValue=None*)

Args:

 extractedValue:

Returns:

`set_knownCauses(extractedValue=None)`

Args:

extractedValue:

Returns:

`set_url(extractedValue=None)`

Args:

extractedValue:

Returns:

`src.software.JIRA.analysisGuide.anyObjectToDictionary(obj)`

Args:

obj:

Returns:

`src.software.JIRA.analysisGuide.getFlattenDictionary(dd, separator=' ',
prefix='')`

Args:

dd: separator: prefix:

Returns:

`src.software.JIRA.analysisGuide.getSuperDictionary(obj=None, debug=False)`

Args:

obj: debug:

Returns:

`src.software.JIRA.analysisGuide.main()`

`src.software.JIRA.analysisGuide.objectToDictionary(objectToWalk)`

Args:

objectToWalk:

Returns:

src.software.JIRA.analysisGuide[unitTest(*options*)

Args:

options:

Returns:

class

src.software.JIRA.jiraSimAnalysis[jiraSimAnalysis(*embeddingData*=None,
embeddingFile: *Optional[str]* = None, *fullDataFile*: *Optional[str]* = None,
embeddingType: *Optional[str]* = None)

Bases: **object**

static SelBest(arr: list, X: int) → list

Args:

 arr: data array X: max number of items to return

Returns: set of X configurations with shortest distance

cosineSimilarityKNN(*data*: *Optional[np.ndarray]* = None,
need2read: *bool* = True, *k*: *int* = 3, *outputFile*: *str* =
'data/output/jiraDataStore/nearestNeighbors.csv', *dimReduce*='SVD')

Args:

 dimReduce: outputFile: (optional) rel path and filename for
 output .csv data: (optional) predefined np.ndarray matrix
 need2read: (optional) boolean to read data if it hasn't been read
 k: (optional) maximum number of nearest neighbors

Returns: Success boolean flag, and a dataframe of k-nearest
neighbors per Jira ID

find_optimal_clusters(*max_k*: *Optional[int]* = None, *plotFile*: *str* =
'ChooseClusterCount.pdf', *max_iter*: *int* = 11, *dataMatrix*=None)

Args:

 dataMatrix: data to analyze, array-like, shape (n_samples,
 n_features) max_iter: max training iterations max_k: maximum
 number of clusters plotFile: file name for plot file

freeData()

Description:

 free data used by all data matrices

gaussianMixClusters(*assertLabels=None*, *labeledDataset=None*,
dimReduce='SVD')

Args:

 dimReduce: choice for dimensionality reduction assertLabels:
 list of associated labels to an assert signature labeledDataset:
 2D-list of jira label-assignments; one list per jira sorted from
 highest to least probable label

Returns:

 groupedIndices: dictionary with key: encoded label number, and
 value: list of Jiras assigned to the label key, mapping: dictionairy
 with key: encoded label number, and value: known cause text
 label

static gmm_js(*gmmP*, *gmmQ*, *n_samples=100000*)

Args:

 gmmP: model P gmmQ: model Q n_samples: number of random
 samples to generate from gaussians

Returns: Similarity score between two Gaussian Mixture Models
trained with disjoint datasets

kMeansClustering(*randState: Optional[int] = None*,
assertLabels=None, *dimReduce='SVD'*)

Args:

 dimReduce: choice for dimensionality reduction assertLabels:
 dictionairy with key: encoded label number, and value: known

cause text label randState: state to start at

Description:

plots clusters using K-Means clustering, and plots sum of squared error per number of clusters

`pickNClustersGMM(max_k: Optional[int] = None, plotFile0: str = 'DistanceBetweenGMMs.pdf', plotFile1: str = 'SilhouetteClusterCount.pdf', plotFile2: str = 'GradientBicScores.pdf', max_iter: int = 11, dataMatrix=None)`

Args:

max_k: max number of clusters plotFile0: filename plotFile1: filename plotFile2: filename max_iter: max training iterations dataMatrix: data to analyze, array-like, shape (n_samples, n_features)

Returns:

`static plotClusterCurve(plotName: Optional[str] = None, dataX=None, dataY=None, xlabel=None, ylabel=None, title=None, yErr=None)`

Args:

xlabel: label for x-axis plotName: name of pdf file dataX: x-axis data dataY: y-axis data ylabel: label for y-axis title: label for title yErr: standard deviation for dataY

Returns:

`static plot_tsne_pca(labels, knownCauses, plotFile: str = 'ClusterPlotPCA.pdf', plotFile2: str = 'ClusterPlotTSNE.pdf', dataMatrix=None)`

Args:

dataMatrix: data to plot, array-like, shape (n_samples, n_features) labels: clusters returned from KMeans knownCauses: speculated cluster labels plotFile: file name for cluster plot PCA plotFile2: file name for cluster plot TSNE

`readData(dataMatrix: Optional[numpy.ndarray] = None)`

Description:

 read embedded data into self.X and store as a sparse.csr_matrix

`reshape3Dto2D()`

Returns: embedding 2D matrix from a 3D matrix-represented set of embeddings for jiras, returns array-like, shape (n_samples, n_features)

`setDataset(csvFile: Optional[str] = None)`

Args:

csvFile: file name to store in self.dataFile member variable

`singularVectorDecomp(need2read: bool = True, numFeatures: Optional[int] = None, numIters: Optional[int] = None)`

Args:

need2read: (optional) boolean to read data if it hasn't been read
 numFeatures: (optional) number of features used to reduce the dimensionality
 numIters: (optional) number of iterations for singular vector decomposition

Description:

 performs singular vector decomposition on a sparse data matrix; if paired with a tfidf term-document matrix, as the embedding method, then the result method is latent semantic analysis

`snippetExtraction(fullDataFile=None, kNNSet=None, outputFile: str = 'data/output/jiraDataStore/nearestNeighborContext.csv', need2write: bool = True)`

`src.software.JIRA.jiraSimAnalysis.main()`

`src.software.JIRA.executeJiraMining.cleanSearchString(searchString: Optional[str] = None)`

Args:

searchString: Assert signature which may potentially contain other noise

Returns: Assert signature string which begins with "ASSERT"

class src.software.JIRA.executeJiraMining.executeJiraMining(datakey: Optional[str] = None)

Bases: **object**

Class-member variables

datakey: the name of the assert to querying Jiras searchkey: actual string to query which uses datakey as base getJiras: reference to classGetJiraIssues outputDir: name of output directory storing Jira data extraced/computed outputDirPath: path of outputDir rawDataDir: directory storing raw crawled data flatDataDir: directory storing flattened data nullRemovedDir: directory storing null-removed data csvDir: directory storing data as csv embeddingDir: directory storing embedding matrix csv labels: reference to list of labels

analyzeJira(embeddedFile: Optional[str] = None, csvInput: Optional[str] = None, outputFile: str = 'data/output/jiraDataStore/nearestNeighbors.csv', embeddingType: Optional[str] = None, embeddingData=None, labeledDataset=None)

Args:

embeddedFile: path to embedding file csvInput: path to csv input file outputFile: path to nearest neighbor file which will be created embeddingType: choice for embedding embeddingData: embedding matrix data structure to use instead of reading in file labeledDataset: list of assigned labels based on similarity comparison between known cause embedding vectors

Returns:

finalTables: dict = known-cause-indexed tables of jiras and their 3-nearest neighbors, tableHeaders: dict = key: encoded label number, value = known cause sentence with % of contained data

`createDataStore(credFile: Optional[str] = None)`

Args:

`credFile`: path to hidden credentials file

Returns:

`embedJira(csvInput: Optional[str] = None, unwantedNoiseJsonFile: Optional[str] = None, embeddingFunction='bert', potentialLabels: Optional[str] = None)`

Args:

`csvInput`: desired csv dataset to utilize
 `unwantedNoiseJsonFile`: desired noise dataset to utilize for noise detection/removal
 `embeddingFunction`: desired embedding to utilize
 `potentialLabels`: associated set of known causes for the assert signature which was used to query Jiras

Returns:

`embedding`: embedding matrix of jiras, `labeledSet`: list of assigned labels based on similarity comparison between known cause embedding vectors

`executeAllJobs(embeddingType: Optional[str] = None, knownCauses: Optional[list] = None, credFile: Optional[str] = None)`

Args:

`embeddingType`: option for choosing which embedding transformation type: ['tfidf' | 'bert']
 `knownCauses`: list of strings of known-causes set wrt the assert signature
 `credFile`: path to .raadProfile/credentials.conf

Returns:

 upon success: (True,

`finalTables`: dict = known-cause-indexed tables of jiras and their 3-nearest neighbors, `tableHeaders`: dict = key: encoded label number, value = known cause sentence with % of contained data)

 otherwise:

(False, None, None)

```
presentNearestNeighborsSetAndContext(embeddedFile: Optional[str]
= None, realDataFile: Optional[str] = None, inputFile: str =
'data/output/jiraDataStore/nearestNeighbors.csv', outputFile: str =
'data/output/jiraDataStore/nearestNeighborContext.csv', need2write:
bool = False, embeddingData=None)
```

```
src.software.JIRA.executeJiraMining.main()
```

jira_issues_lists class converts the fields read from the JSON file into a list for easier consumption. Also contains common helper functions.

```
class src.software.JIRA.classJiraIssuesList.issue_utility
```

Bases: **object**

```
# A set of helper function for accessing information about a issue # <p>
```

```
getLastToDate(aIssue, stateToMatch)
```

```
# Returns the last date that a issue transistioned to the matching state
# # <p> # # @param aIssue A single issue # @param stateToMatch
string of the state transition to match #
```

```
class src.software.JIRA.classJiraIssuesList.jira_issues_lists
```

Bases: **object**

```
# jira_issues_lists class converts the fields read from the JSON # file
into a groovy list for easier consumption by calling # routines. # <p>
```

```
addIssue(theIssue)
```

```
# Adds the Jira information of interest to the list. # Note: This is
much different than the groovy version. # The statistic gathering
routines have been removed too. # # <p> # # @param theIssue the
class that parsed a single line of the JSON #
```

```
inBCH = ['Blocker', 'Critical', 'High']
```

```
inClosed = ['Validation Complete', 'Closed']

inDevVal = ['Assigned Development', 'Assigned Validation', 'Confirmed Defect', 'In Progress', 'New', 'Reopened', 'Groomed']

inFPV = ['Fixed Pending Validation']

inFixed = ['Duplicate Pending Validation', 'Fixed Pending Execution', 'Fixed Pending Validation']

inHold = ['On Hold']

issueList = []

programDict = {}

reset()
    # Resets and clears all of the data # Note: likely obsolete function
    #

setScrumTeam(aScrumTeam)
    # Sets the scrum team that the report is being made for. # <p> # This
    # allows us to filter the results based off of a single scrum team. # #
    @param aScrumTeam the scrum team name #

setSprintStartDate(startDate)
    # Records it given start date. # <p> # Note: We calculate back to the
    # start, since initially the scripts were # used to summarize the
    # previous sprint. Going forward, the scripts # are fed the sprint end
    # date for the commits and calculate backward from # there to get the
    # start date to look for issues. # # @param startDate The end date of
    # the sprint (data type of Date) #

setSprintStartDateFromEnd(pastSprintEndDate)
    # Takes the spring end date, calculated the start date, and records it.
    # <p> # Note: We calculate back to the start, since initially the
    # scripts were # used to summarize the previous sprint. Going
    # forward, the scripts # are fed the sprint end date for the commits and
    # calculate backward from # there to get the start date to look for
```

```
issues. ## @param pastSprintEndDate The end date of the sprint  
(data type of Date) #  
  
teamIssueList = []  
  
test()  
    # Adds the Jira information of interest to the list. # Note: This is  
    much different than the groovy version. # The statistic gathering  
    routines have been removed too. ## <p> ## @param theIssue the  
    class that parsed a single line of the JSON #  
  
whichScrumTeam = "  
  
Handles all of the generations of the HTML files. (Tables, graphs, etc.)  
  
class src.software.JIRA.classHtmlGen.htmlGen  
Bases: object  
  
# Generates all of the html output # <p>  
  
BeginHtlmOutput(title, heading1)  
    # Write everything found at the beginning of an HTML file up to the  
    body section. ## @param title Title to be displayed in the  
    tab/window section # @param Heading1 First large heading in the  
    HTML document #  
  
BeginTable()  
    # Tag to start HTML table  
  
EndHtlmOutput()  
    # EndHtlmOutput # Close the body and html document  
  
EndTable()  
    # Tag to end HTML table  
  
RawHtlmOutput(rawHtml)
```

```
# Write raw information to the HTML file. It puts into whatever  
string you send. ## @param rawHtml String to place in the HTML  
file
```

TableData(*theIssue, theColumns, keyColmn, firstColumnColor, title*)

```
# TableData # Write the information in each cell of the table. This  
table only allows 1 color change ## @param theIssue The issue or  
the list of information to display ## @param theColumns Which  
elements of the list to use ## @param keyColmn Which element has  
the issue key in it, so that a hyperlink can be provided to the issue ##  
@param firstColumnColor The color, if any, of the first column ##  
@param title HTML title attribute to mouse over inforamtion
```

TableDataColor(*theRow*)

```
# Write a row of the table. Color can change per cell, all columns  
sent will be displayed, and there is a specific format per each ##  
## @param theRow A list of dictionary entries with the following  
information, "text", "key", "color", "title"
```

TableHeader(*theColumns, theWidths*)

```
# TableHeader # Write a header row in a table. Assumes table  
already started. ## @param theColumns Text to display in each  
column (a list) ## @param theWidths The percentage of the width for  
the column (a list)
```

googleGauge(*title, gaugeValue, gaugeMax, yellowFrom, yellowTo,*
redFrom, redTo)

```
# Generate a pie chart using the google graph API ## @param title  
Title to appear on gauge ## @param gaugeValue Value that the  
pointer points to on the guage ## @param gaugeMax Maximum  
number on gauge ## @param yellowFrom Value where yellow starts  
## @param yellowTo Value where yellow ends ## @param redFrom  
Value where red starts ## @param redTo Value where red ends ##
```

googlePieChart(*title, idName, data*)

```
# Generate a pie chart using the google graph API # Note: The input  
is different than the Groovy version, which uses lists instead of  
dictionaries ## @param title The issue or the list of information to  
display ## @param idName The graph ID ## @param data Which  
element has the issue key in it, so that a hyperlink can be provided  
to the issue
```

```
googleStackedColumn(title, idName, headerList, dataList,  
columnColors)
```

```
# Generate a pie chart using the google graph API ## @param title  
The issue or the list of information to display ## @param idName  
The graph ID ## @param headerList Label for each piece of the data  
to be displayed ## @param dataList List of the data to be displayed ##  
@param columnColors Colors of the columns
```

```
htmlOutput = None
```

```
htmlOutputFileName = ''
```

```
openHtmlOutput(filename)
```

```
# openHtmlOutput # Open a file for writing. Delete the file if it  
exists ## @param filename Filename to open #
```

```
urlRoot = 'http://nsg-jira.intel.com'
```

Description: Get Jira Issue class performs steps need to query and retrieve issues from jira

Sample Usage for Jira data-preprocessing:
- Working Directory: raad
- command: python3 src/software/JIRA/classGetJiraIssues.py -k DF049 -a Data/jiraData/queries.txt

- flags: -k: datakey, -a: all, perform all functions: query data -> output clean data

Construct the REST call (url to call) to get the information that is needed
REST API documentation can be found here:

<https://developer.atlassian.com/display/REST/REST+API+Developer+Documentation>

Examples of types of queries can be found here

<http://extensions.xwiki.org/xwiki/bin/view/Extension/JIRA+REST+Integration>

<http://docs.atlassian.com/jira/REST/latest/>

In general, for 4.3.1 the format follows: \${urlRoot}/rest/api/2.0.alpha1

In general, for 5.1 the format follows: \${urlRoot}/rest/api/2/

To perform the same type of queries that are used in the Jira GUI, the following is used:

 \${urlRoot}/rest/api/2.0.alpha1/search?jql=

For clarity, the query has been separated onto a different line. This makes cut and paste from existing queries a lot easier.

Examples of queries:

```
addr = "${urlRoot}/rest/api/2/issue/$issueNumber?  
expand=changelog,schema" addr = "${urlRoot}/rest/api/2/issue/NSGSE-  
8413?expand=changelog,schema" addr = "${urlRoot}/rest/api/2/search?  
jql=issuekey=NSGSE-8413&expand=changelog,schema"
```

```
class src.software.JIRA.classGetJiraIssues.get_jira_issues(username=None,  
password=None, debug=False, defaultPasswordFile: Optional[str] =  
None, datakey: str = 'None')
```

Bases: **object**

```
# get_jira_issues class retrieves the issues from Jira and stores them in a  
JSON format # <p> # The following are performed by this class # -  
password input (username is hard coded to daescamx # -  
Authentification with Jira Sever # - Retrieval of issues matching the  
query 40 at a time # - Retrieval and storage of detailed information for  
each of the matching issues # # Note: At this time, the detailed
```

information for each issues can only be retrieved by # querying each issue individually. I have not been able to find a way to retrieve # it differently. # <p>

aIssuesList = <*classJiraIssuesList.jira_issues_lists object*>

aJiraIssue = <*classJiraIssues.jira_issue object*>

aSupportFile = <*classSupportFiles.support_file object*>

addr = 'https://nsg-jira.intel.com/rest/api/2/search?jql='

allIssues = []

atlassianJira(*query: Optional[str] = None*)

Description: Issues query for JIRAs based on query parameter

Args:

query: Formulated query string using JQL, then passed to the Jira urlRoot

Returns: json response containing raw data matching the query

authString = ''

static

createCSVFromJson(*ifile='data/output/jiraDataStore/removeNULL/removeNullASSERT_DE003.json', ofile='data/output/jiraDataStore/csv/jsonToCsv.csv', fieldsFile='./data/jiraUtilMeta/fieldsMapping.csv', unwanted: Optional[set] = None*)

Args:

ifile: input FLAT json file ofile: output name fieldsFile: name of file containing desired newline separated JIRA fields, which are used as columns unwanted: set of unwanted features (i.e. noise)

Returns:

- When creating csv, need to get dictionary per complex field, then handle the dictionary comparison when

doing sentence tokenization for word-embedding matrix generation

`findMostRecentUpdate(updateFilename)`

Searches through all of the issues to find the most recent last updated. Currently will return the beginning of the day of the last update # # @param updateFilename The JSON file to compare the query issues against.

`static flattenJson(jsoninput=None, jsonoutput: Optional[str] = None)`

Description: flatten json input file

Args:

`jsoninput`: input for flattening `jsonoutput`: flattened output

`jiraCollect(queryStr: Optional[str] = None, rawDataPath: str = 'data/output/jiraDataStore/raw', flatDataPath: str = 'data/output/jiraDataStore/flat', nullRemovedPath: str = 'data/output/jiraDataStore/removeNull', csvPath: str = 'data/output/jiraDataStore/csv', noiseTextPath: str = './data/jiraUtilMeta')`

Description: Issue a JQL query and perform all preprocessing

Args:

`noiseTextPath`: location of utility data used to reference which Jira fields are considered noise `csvPath`: location of where csv datasets will be stored `nullRemovedPath`: location of where intermediate null-removed datasets will be stored `flatDataPath`: location of where intermediate flattened datasets will be stored `rawDataPath`: location of raw crawled json data `queryStr`: String of text to pass as query for Jira searching

`jiraLogin()`

Gathers the login information for Jira. # This allows us to filter the results based off of a single scrum team.

`jqlStr = ''`

```
keyAndUpdated = {}

outputFilename = ""

static removeNullFields(filename,
outfile='data/jiraData/removeNull/removeNull')

Args:
    filename: filename of flat json object outfile: ouptut file name

retrieveIssueDetails(issueKeyList)
    ## Retrieve issues all of the information for each issue in the list
    passed in ## @param issueList Issue to retrieve detailed
    information from in list form ## @returns Returns the received json
    as a list of strings

retrieveIssueList(query)
    ## Retrieve issues matching the query 40 at a time # Populates the
    dictionary of key and updated with that information ## @param
    query query to use to retrieve issue summaries #

sendRestRequest(url)
    ## Connects to the url and returns raw response # Note: Assumes
    authorization string has already been set by jiraLogin ## @param
    url the entire url and request information ## @return - the raw
    response from the server #

setDatakey(name='None')
Args:
    name: Name of content (otherwise thought of the label of the
    data search query

Returns:

setOutputFilename(filename, extension='json', mode=1)
    Sets the output filename for the JSON Jira results and renames the
    current file to the highest available archive number
```

Args:

filename: arbitrary name for a file extension: file format mode:
new methods use mode 2, any other value provides original
behavior

Returns:

the new archive name that replaced the existing one

updateSnapshot(*updateFilename, scrunTeamsToKeep*)

Compares the updated date of the issues read from jira (in
keyAndUpdate) to those in the snapshot (aIssuesList.issueList) #
Issues are removed from the update list if the snapshot contains ones
with the same or more recent date. # Each issue remaining in the
update list is retrieved and added to the snapshot, with all non-
updated issues added afterwards # # ** WARNING ** # This
function expects the snapshot to already have been read in by
findMostRecentUpdate # # @param updateFilename The JSON file
to compare the query issues against. # @param scrunTeamsToKeep
A list of the scrum team issues to keep in the database #

urlRoot = 'https://nsg-jira.intel.com'

static writeFile(*data, fileformat, filepath*)

src.software.JIRA.classGetJiraIssues.main()

src.software.JIRA.classGetJiraIssues.queryJira(*_getJiraIssues, qry*)

Args:

_getJiraIssues: classGetJiraIssues object *qry*: query string

Returns:

src.software.JIRA.jiraEmbedding.distFuncTest()

src.software.JIRA.jiraEmbedding.distanceMeasure(*contextA*:
Optional[numpy.ndarray] = None, *contextB*: *Optional[numpy.ndarray]* =
None, *evaluationFunction*=<function *pearsonr*>)

```
class src.software.JIRA.jiraEmbedding.jiraEmbedding(csvInputFile:  
Optional[str] = None, distanceFunction=None, numClusters: Optional[int]  
= None, unwantedNoiseFile: str = 'data/jiraUtilMeta')
```

Bases: `object`

`bertEmbedJira(labels=None)`

Args:

labels: set of known causes associted with the assert signature

Returns:

docFields: embedding 3D matrix, where each jira is a matrix of
 jira-field embeddings, *labels*: labeled jiras in order of how they
 are organized in the original csv dataset

`getFieldDocString(row, noiseSet: set)`

Args:

row: a row in the dataframe of jiras, each row is a jira
 noiseSet: set of unwanted jira fields which we do not tokeinze

Returns: *returnDoc*: string of all text in jira,

fields: list of strings where each element is the text for a field

`getFieldDocument(row, noiseSet: Optional[set] = None)`

`getFieldMessege()`

Returns:

documents: list of size N where N is the number of data-points
 in the csv dataset;
 each *n_i* is a string containing all of the data-point's text

docFields: 2-D list with N rows and L columns; N is the number
 of data-points in the csv dataset;

 each *n_i* is itself a list of size L where L is the number of
 features per data-point; each *n_ij* is a string containing the
 text for feature j of data-point i

labelNeededIndices: indices for columns 'summary',
'description', 'Resolution Description', 'Fixed in Component'

getFieldSet()

static getJiraLabels(docs=None, jiraLabels=None, model=None, relevantFields=None)

Generator function which computes the accumulated similarity between the embedded label vector, and the vectors for the fields in relevantFields. It then generates the highest scoring/matching label.

Args:

docs: embedding 3D matrix, where each jira is a matrix of jira-field embeddings
jiraLabels: labeled jiras in order of how they are organized in the original csv dataset
model: model used to create embeddings
relevantFields: indices for field columns: 'summary', 'description', 'Resolution Description', 'Fixed in Component'

Returns: (label, accumulated similarity score) of most similar label per Jira

getSentenceString(inlist, d: Optional[dict] = None, skipList: Optional[set] = None)

Recursive function to grab text values from a dictionary of dictionaries

Args:

inlist: list containing previously populated text for a field
d: dictionary item which we traverse and grab text from
skipList: fields to skip

Returns: result string of text after walking down to the deepest level of the dictionary of dictionaries

getSentences(inlist: Optional[list] = None, d: Optional[dict] = None, skipList: Optional[set] = None)

`getTokenizedDocs()`

Returns:

documents: 2-D list of N rows; N corresponds to the number of data-points in the dataset;
each `n_i` is the list of tokens (i.e. words) per data-point

`tfidfVectorization(outputName: Optional[str] = None, writeData: bool = True)`

Args:

`writeData`: option to allow for writing of data to file
`outputName`: name for output csv of embedded data

`src.software.JIRA.jiraEmbedding.main()`

`src.software.JIRA.jiraEmbedding.placeholder(doc)`

`src.software.JIRA.jiraEmbedding.setPairScore(contextA: Optional[dict] = None, contextB: Optional[dict] = None, evaluationFunction=<function pearsonr>, debug: bool = False)`

description:

computes the intersection of keys between `contextA` and `contextB`, and then for each of the keys in the intersection, compute the similarity scoring based on the parameter function.

Args:

`contextA`: dictionary of a particular nonuniform metadata
`contextB`: dictionary of a particular nonuniform metadata
`evaluationFunction`: Similarity score function pointer
`debug`: WHEN TRUE, the function will use double the memory for subobjects,

and push more data onto the stack; this can lead to a segmentation fault/memory-leak for garbage collection (GC). WHEN FALSE, the function will inplace-update subobjects and can potentially still push data onto the stack due to the recursive nature of the function.

Returns:

sim: cumulative similarity score of the dictionary key-value
intersection dictofSims: dictionary of key to similarity scores found
for intersection intersectionList: intersection of keys of two
dictionary sets excludedListA: list of keys not in A excludedlistB:
list of keys not in B

Complexity:

Theta: TBD Big-O: $f(n, k) = 4n + n*k$

$$k = f(k, l) \quad l = f(l, m) \quad f(n, f(n, n')) = 4n + n(4n + n')$$

$$= 4n(1 + (1/4n)(n + n')) \text{ assumption: } n = n'$$

$$\begin{aligned} & \lim(n=0, n'=\inf) | 4(\inf) + \inf(4(\inf) + \inf) \\ &= 4q + q(4q + q) = 5q^2 + 4q = O(q^2 + q) \end{aligned}$$

assumption: $n < n'$

$$\begin{aligned} \text{interval: } & [n' \leq n'', n' > n''] \\ &= q'' \wedge (q') \end{aligned}$$

$$f(n, k) = [n^2 + n, n^{(n^2 f(n, k))}]$$

$$\Omega(f) = n \quad \Theta(f) = n^2 + n \quad \text{Big-O}(f) = n^2$$

`src.software.JIRA.jiraEmbedding.validNumericalFloat(inValue)`

`classSupportFiles` Reads the tab delimited support files. Provides access to the data within those files Currently support files are of the following types:

Milestones Commits

The expected column format for each of the supported file types is described in the get function for it.

`class src.software.JIRA.classSupportFiles.support_file`

Bases: `object`

```
# parses the support files and provides easy access to the data therein #
<p> # Note: Dependent on commits being entered in a fixed format as
defined for each support file. ## @version 1.0 #

asksList = []
commitColor = 'CC99FF'
commitList = []

getCommitFile(filename)
    # getCommitFile ## Performs all the functions to get the commitfile
    data and parse it into a easy to use form # <p> ## @param filename
    filename to parse

getMilestoneFile(filename)
    # getFile ## Performs all the functions to get the milestonefile data
    and parse it into a easy to use form # <p> ## @param filename
    filename to parse

issueDictionary = {}

milestoneList = []
scrumTeam = ""
sprintEndDate = ""
stretchColor = 'F0E0FF'

tabDelimitedToList(filename, expectedColumns)
    # tabDelimitedToList ## Reads the tab delimited commits file and
    converts it to a list # <p> ## @param filename filename to parse #
    @param expectedColumns the expected columns to read. also used
    to name those columns

threshold = []
```

class
src.software.container.basicTypes.TelemetryData(*typeOfTelemetry=None*,
filename=None, *path=None*)

Bases: **object**

Telemetry: Telemetry information in respect the the current payload
Internal: Telemetry Firmware version is the internal tracking across all
products.

Major: [1 to M] Minor: [0 to K]

Type: ["Host", "Controller"] - Host or controller initiated telemetry
asynchronous command. Snapshots: Telemetry binary payloads names
and path pairs.

Filenames: "PHAB8506001G3P8AGN_sample.bin" - File name on
the host system. Paths:
"C:/Source/SSDDev/NAND/gen3/tools/telemetry/sample" - Path
on the host system to binary. PayloadValidity: ["Verified",
"Unknown", "Corrupted"] creationTime: [year, month, day[, hour[,
minute[, second[, microsecond[, tzinfo]]]]]] - Represents the time
at which the data object was imported to the tool.

getWalkDictionary()

getWalkList()

class src.software.container.basicTypes.UID_Application(*uid=None*,
major=None, *minor=None*, *data=None*)

Bases: **object**

Meta Description - The meta data for the data object.

VersionTracking(object): Version information. Class Object - The class
used.

getData()

getMajor()
getMinor()
getUID()
getWalkList()
setData(*objectIn=None*)
setMajor(*objectIn=None*)
setMinor(*objectIn=None*)
setUID(*objectIn=None*)

class src.software.container.basicTypes.UID_Data(*uid=None, major=None, minor=None, data=None*)

Bases: **object**

Meta Description - The meta data for the data object.
VersionTracking(object): Version information. Data Object - The data payload with defined content.

getData()
getMajor()
getMinor()
getUID()
getWalkList()
setData(*objectIn=None*)
setMajor(*objectIn=None*)
setMinor(*objectIn=None*)

`setUID(objectIn=None)`

`class src.software.container.basicTypes.UID_Org(vendor='Intel',
bussinessUnit='NSG', group='STG', team='AMPERE')`

Bases: `object`

Organization - Identification of the source-destination of the data.

Vendor - High level organization. Bussiness Unit - Individual business group within the organization. Group - Group within the bussiness unit. Team - Team whithin the group.

`getGroup()`

`getTeam()`

`getVendor()`

`getWalkList()`

`getbusinessUnit()`

`setBusinessUnit(objectIn=None)`

`setGroup(objectIn=None)`

`setTeam(objectIn=None)`

`setVendor(objectIn=None)`

`class src.software.container.basicTypes.VersionTracking(otype=None,
uid=None, major=None, minor=None)`

Bases: `object`

Information to track the version of any generic object. otype: type of object to identify. UID: unique identifier. Major: Represents a unique structure organization of a given object. Minor: Represents an extension of an object to the end of the current payload without any additional changes.

getMajor()
getMinor()
getType()
getTypeHuman()
getUID()
getWalkList()
setMajor(*varIn=None*)
setMinor(*varIn=None*)
setType(*varIn=None*)
setUID(*varIn=None*)

src.software.container.basicTypes.main()

Performs the auto parsing of data control to generate telemetry definitions within a python c-type for valid structures.

src.software.container.basicTypes.uniqueIdentifierTypes_e(*otype=None*)
unique identifier lookup function Args:

otype: str, int

Returns: if is int then return string; otherwise, if the type is string return int

class src.software.container.indirection.OneAPI(*object=None*)

Bases: **object**

Class for accessing One API for storage data containers.

class Indirection(*object*)

Bases: **object**

A dictionary which can lookup values by key, and keys by value. All values and keys must be hashable, and unique.

class Lookup(Indirection)

Bases: **object**

A dictionary which can lookup values by key, and keys by value. The lookup method returns a list of keys with matching values to the value argument.

`lookup(value)`

class Reflect(object)

Bases: **object**

Construction of a dynamic list by reflecting the order by swapping the context (reversing). A dictionary which can lookup values by key, and keys by value. All values and keys must be hashable, and unique.

`dictionaryForward()`

`dictionaryReverse()`

`forwardDirectionDict = None`

`getByteSize()`

`getName()`

`getUID()`

`getVersion()`

`reverseDirectionDict = None`

`uidObject = None`

creationTime = *None*
protocolIdentification = *None*
sourceContext = *None*
telemetryBinary = *None*
telemetryVersion = *None*

src.software.container.indirection.main(*usage*)

Performs the auto parsing of data control to generate telemetry definitions within a python c-type for valid structures.

Brief:

Utility function to template all python files.

Description:

Class library is designed to be used as a tracking template file to ease use across all files and ensure tracking.

Layout `__init__.py`

```
<Text Template> <import items> myNameIdentify =
templateUtility("C:/Path", "__init__.py", 1, 0, "04-09-2020
11:58:00", "") myNameIdentify.uInit(self) <Other Code>
```

Layout `myName.py`

```
<Text Template> <import items> myNameIdentify =
templateUtility("C:/Path", "myName.py", 1, 0, "04-09-2020
11:58:00", "") myNameIdentify.uPy2toPy3Convert()
myNameIdentify.uImportDirectoryTree(True,False)
myNameIdentify.uEXE() <Other Code> myNameIdentify.uMain()
```

```
class src.software.utility.templateUtility.templateUtility(absPath=None,
filename='unknown.py', major=0, minor=0, time=datetime.datetime(2022,
6, 29, 18, 37, 52, 452077), user='')
```

Bases: `object`

`templateUtility` are documented in the same way as classes.

The `__init__` method may be documented in either the class level docstring, or as a docstring on the `__init__` method itself.

Either form is acceptable, but the two should not be mixed. Choose one convention to document the `__init__` method and be consistent with it.

Note:

Do not include the `self` parameter in the `Args` section.

Args:

`msg` (str): Human readable string describing the exception. `code` (int, optional): Error code.

Attributes:

`msg` (str): Human readable string describing the exception. `code` (int): Exception error code.

`maxName = 16`

`maxPath = 256`

`maxTime = 32`

`printer()`

`uEXE()`

`uImportDirectoryTree(doIt=True, debug=False)`

`uInit()`

`uMain()`

`class src.software.DP.preprocessingAPI.preprocessingAPI`

Bases: `object`

Utility class for handling data preprocessing tasks

static anyObjectToDictionary(self)

function for turning an object into a dictionary using the `__dict__` field

Args:

self: object of interest

Returns:

Dictionary representation of self - object of interest

static classToDictionary(objectToWalk, filterData=True)

function for turning a class into a dictionary using iteration and the wrapper function `vars()`

Args:

objectToWalk: object to be turned into dictionary
filterData: Boolean value to ignore attributes that start with underscore

Returns:

Dictionary representation of self - object of interest

*flattenJson(jsoninput, dataID,
jsonoutput='data/jiraData/flat/flatIssues.json')*

Args:

jsoninput: String for the file name of the input for flattening
dataID: String for the ID of the data stored in the json object
jsonoutput: String for the file name of the flattened output

Returns:

data: flatten dictionary containing the data stored in the jason file

generateMP(dataDict, subSeqLen=20, debugStatus=False)

function for generating a matrix profile for multiple time series contained in dataDict

Args:

debugStatus: Boolean value for the debug status
dataDict: Dictionary containing all the time-series data
subSeqLen: Integer value for the length of the sliding window to be used to generate the matrix profile

Returns:

MP: Dictionary containing all the data for the matrix profiles associated with the given fields of the specified objects

static getDateFromName(dirName='sample')

function to extract datetime object from directory name

Args:

dirName: String representation of the directory name

Returns:

datetime object containing the date parsed in the dirName

static loadCSVIntoDict(configFileName, debug=False)

function that loads the CSV object values into a dictionary

Args:

configFileName: String of the path name for the csv file containing the data for the time-series
debug: Boolean flag to activate debug statements

Returns:

Dictionary with the object values from the ConfigParser

static loadConfigIntoDict(config, debug)

function that loads the ConfigParser object values into a dictionary

Args:

config: ConfigParser that contains the object values
debug: Boolean flag to activate debug statements

Returns:

Dictionary with the object values from the ConfigParser

loadDataDict(configFilePath, debugStatus=False)

function for loading all datas from a the configuration file into a dictionary

Args:

configFilePath: String for the file path containing the .ini file to be processed
debugStatus: Boolean flag to activate debug statements

Returns:

intermediateDict: Dictionary containing all the timeseries data

static loadDictIntoConfig(config, resultDict)

function that loads the previous objects values into a new dictionary

Args:

config: ConfigParser that will contain the object values
resultDict: Dictionary containing the object values to be transferred to config

Returns:

static objectToDictionary(objectToWalk)

function for turning an object into a dictionary using iteration

Args:

objectToWalk: object to be turned into dictionary

Returns:

Dictionary representation of objectToWalk

populateMPStruct(MP, subdict, object_t, subSeqLen)

function for generating the dictionary with the matrix profile values for the time-series

Args:

MP: Dictionary where all the data lists will be stored subdict:
Dictionary where the unprocessed data lists are contained
object_t: String for the name of the object for which we are
extracting the matrix profiles (ex. uid-6) subSeqLen: Integer for
the window size to be used in the matrix profile generation

Returns:

static readFileIntoConfig(configFileName)

function for reading a file into a config

Args:

configFileName: String of the path name for the configuration
file containing the data for the time-series

Returns:

config object with the data of the file already inside it

*removeNullFields(filename, dataID,
outfile='data/jiraData/removeNull/removeNull.json')*

Make sure that JSON is flatten

Args:

filename: String for the filename for flat json object dataID:
String for the ID of the data stored in the json object outfile:
string for the name of the output file

Returns:

setOutputFilename(filename, extension='json', mode=1)

Sets the output filename for the JSON Jira results and renames the
current file to the highest available archive number

Args:

filename: arbitrary name for a file extension: file format mode:
new methods use mode 2, any other value provides original

behavior

Returns:

the new archive name that replaced the existing one

static transformDict(intermediateDict, debug)

function for transforming the dictionary from a flat structure into the nested structure required for generating the right format for the plain text files

Args:

intermediateDict: Dictionary containing the flat structure for all the objects contained in the configuration file
debug: Boolean flag to activate debug statements

Returns:

Dictionary containing the expanded structure for all the objects contained in the configuration file

class src.software.datacontrol.dataControlGenMain.DatacontrolGen(file)

Bases: **object**

compareDicts(local, remote)

compareObjs(remote)

Could be used to automatically resolve local and remote datacontrol.h merge conflicts

class

src.software.datacontrol.dataControlGenMain.DatacontrolGenCSV(file, tempMetaRepo=None)

Bases:

[src.software.datacontrol.dataControlGenMain.DatacontrolGen](#)

compareDicts(local, remote)

compareObjs(remote)

Could be used to automatically resolve local and remote
datacontrol.h merge conflicts

`create(tempUid, structName, dataArea, dataGroup, product)`

Similar to edit, but creates new uid if temp and if not previously defined. Call BEFORE creating FW code to ensure compatibility of uid with code.

`deprecate(uidNum, version)`

Deprecating is merely a state where there is no product/subbuilds information for a structure.

`edit(uidNum, currVersion, autoparsability=None, major=None,
minor=None, persistence=None, dependency=None,
securityClass=None, byteSize=None, dataArea=None,
structDup=None, dataGroup=None, structName=None, size=None,
withinAssert=None, product=None, domain=None, owner=None,
description=None)`

Makes local changes only. Call implement to push changes to remote database

`genLocalPythonDict()`

For local use for TWIDL tstructor and dynamic structures

`getAvailableTemp()`

Parse list of UID's to find next available UIDs. It is vital that UIDs, even if deprecated, never get reused, for consistency across versions and appropriate tagging.

`getDict()`

`getDicts()`

`implement()`

Responsible for assigning uids to newly defined structs in CSV file and editing uid meta information in remote database to include

newly created uids.

`readFile()`

Reads CSV file to create uid objects

`uidInitByList(list)`

Performs List Validation. Whenever creating UIDs, ensure Input Validation

`updateFile()`

`updatePythonDict()`

Function to initialize uid python dicts in REMOTE location

`class src.software.datacontrol.dataControlGenMain.DatacontrolGenH(file,
overwrite=None)`

Bases:

[src.software.datacontrol.dataControlGenMain.DatacontrolGen](#)

`compareDicts(local, remote)`

`compareObjs(remote)`

Could be used to automatically resolve local and remote datacontrol.h merge conflicts

`convertToH(dict)`

`matchWithCSV(csvObject)`

`readFile()`

Extracts telemetry uid objects from datacontrol.h Pass in no argument to automatically search for local datacontrol.h, pass in argument for master datacontrol.h location

`updateFile()`

```
class src.software.datacontrol.dataControlGenMain.UID(uidNum,  
structName, dataArea, dataGroup, version='2.0', implemented=False,  
deprecated=False, temp=False)
```

Bases: `object`

`varConstructor()`

```
class src.software.datacontrol.dataControlGenMain.cUID(uidNum,  
structName, dataArea, dataGroup, product, autoparsability='', major=0,  
minor=0, persistence='unknownPersistence',  
dependency='unknownDependency',  
securityClass='unknownClassification', byteSize='',  
structDup='unknownDuplication', size='', withinAssert='',  
domain='unknownDomain', owner='', description='', autoParse='No',  
version='2.0', implemented=False, deprecated=False, temp=False)
```

Bases: [`src.software.datacontrol.dataControlGenMain.UID`](#)

`asDict()`

`varConstructor()`

```
src.software.datacontrol.dataControlGenMain.dataControlGenWrapper(out  
put=<_io.TextIOWrapper name='<stdout>' mode='w' encoding='utf-8'>,  
to_create=None, to_deprecate=None, to_edit=None, implement=None,  
repodir=None, repoRoot=None, file=None, debug=False, update_h=False,  
update_dict=False)
```

Wrapper builds dataControlGen instance, executes and cleanup

```
class src.software.datacontrol.dataControlGenMain.hUID(uidNum,  
structName, dataArea, dataGroup, autoParse='No', line=None,  
version='2.0', implemented=False, deprecated=False, temp=False)
```

Bases: [`src.software.datacontrol.dataControlGenMain.UID`](#)

`varConstructor()`

```
src.software.datacontrol.dataControlGenMain.main()
```

src.software.datacontrol.dataControlGenMain.remove_readonly(*func*, *path*,
excinfo)

src.software.datacontrol.dataControlGenMain.sizeInDec(*sizeInHex*)

src.software.datacontrol.dataControlGenMain.uidSort(*uidNum*)

src.software.datacontrol.dataControlGenMain.uidTempSort(*uidNum*)

Brief:

intelTelemetryParser.py - Generic parser definitions for parsing
telemetry data object blobs

Description:

This file contains the base class and function definitions needed to build
a parser for a telemetry data object

Classes:

Enter GetHeaders("parsersintelTelemetryDataObject.py") to display
Class listings.

class
src.software.parse.intelTelemetryDataObject.DataParserV2_0(*expectedId*,
filename, *moduleName=None*, *baseStructName=None*,
ignoreSizeMismatch=False)

Bases: **object**

Brief:

ExampleDataParser() - Data parsing example code.

Description:

Example code for parsing a data object.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

FillDataStructure(*moduleName*, *baseStructName*, *includeMinor=True*,
ignoreSizeMismatch=False)

getFile()

getObjectSize()

getVersionName(*includeMinor=True*)

class

src.software.parse.intelTelemetryDataObject.intelTelemetryDataObjectHeaderV2_0_struct

Bases: `_ctypes.Structure`

Brief:

intelTelemetryDataObjectHeaderV2_0_struct() - Intel Telemetry object header structure definition.

Description:

Intel data object header structure definition.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

`cpuId`

Structure/Union member

`getCustomerId()`

`getIdNumber()`

`getMajorVersion()`

`getMinorVersion()`

`getVersionName(includeMinor=True)`

`idNumber`

Structure/Union member

`majorVersion`

Structure/Union member

`minorVersion`

Structure/Union member

`reserved`

Structure/Union member

`tostr()`

class

`src.software.parse.intelTelemetryDataObject.intelTelemetryDataObjectHeaderV2_0_union(imageBuffer=None)`

Bases:

[`src.software.parse.intelTelemetryDataObject.telemetryStruct_union`](#)

Brief:

`intelTelemetryDataObjectHeaderV2_0_union()` - Intel Telemetry object header union fill structure.

Description:

This class extends telemetryStruct_union to fill a data object header structure.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

Bytes

Structure/Union member

getSize()

getStruct()

header

Structure/Union member

class

```
src.software.parse.intelTelemetryDataObject.telemetryStruct_union(image
Object=None, maxBytes=None, ignoreSizeMismatch=False,
imageBuffer=None, file=None, returnBlockSize=None)
```

Bases: `_ctypes.Union`

Brief:

`telemetryStruct_union()` - Union to load structure from file data or array.

Description:

Fill union with data from a binary file or other array object.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

bufdata implements interface(s) for working with Twidl buffers; Primarily used and maintained by NSG's Test Firmware Team.

Please talk to CS or the current maintainer before making modifications to this file.

class src.software.parse.bufdata.BufData

Bases: **object**

BufData - Abstract class for data structures that move information to/from Twidl buffers. Due to the "Figure out what to do at runtime" nature of python, it's not necessary to implement an interface, save to have the inheriting classes break constantly and/or reference this doc work. Inheriting from this class helps developers in TFW Twidl to code in a consistent fashion.

BufData classes should NOT be responsible for drive interface or population of the buffers they will work with without a well documented reason to do so. The reason for this are: - Tools to populate the buffer may be volatile (Twidl, Firmware), the

tools to manipulate the data from the drive need not be.

- As BufData classes are able to work from files, it is possible and desirable to work with them in a Non-Twidl Environment. This includes but is not limited to debug away from drives and data analysis of information from the manufacturing process.

`append_to_file(file_name, offset=0)`

Saves additional information to 'file_name' in binary format.

Implementation compatible with 'to_file'.

Most bufdata/dataparse implementations won't make much use of this. The only known one so far is burn-in log, which is simply an endless stream of repeating elements: log records.

`bits_different(other)`

Compares this bufdata structure to 'other'.

Returns the number of bits that differ between the 2.

Raises Exception if similar() is false. There's no good way to report that the comparison could not be made.

Check similar if you're concerned about breaking in this manner

Note that this can be a very time-expensive operation.

`byte_size()`

Return the number of bytes (int) this uses in a buffer.

<Historical Note> Previous implementation in TFW Twidl set this as an attribute; will require refactor as it's implemented. Adoption of PEP 8 naming conventions by this class do the same.

`file_info()`

Returns a sting containing the file name, created datetime, and file size,

returns " if object never accessed a file.

`from_buf(myBuf, offset=0)`

Extracts information from 'myBuf' starting at 'offset'.

myBuf contains an iterable + indexable object, typically a Twidl buffer, which contains the information to be parsed.

offset is the location from which the class will start to extract data.

It is typical to see this optionally called from `__init__`, especially on classes that extract data that is read to a buffer from drive

<Historical Note> Parameter order is inverted from old TFW Twidl code; will require refactor as it's implemented. Adoption of PEP 8 naming conventions by this class do the same.

`from_file(file_name, offset=0)`

Loads binary information from 'file_name'.

Implementation compatible with 'to_file'.

`isAllFFs()`

checks to see if all the data as represented in the buffer is all 0xFF.

Useful, as erased nand is 0xFF

`last_file = None`

`last_file_stat = None`

`name = None`

`similar(other)`

Check if....

-'other' is 'None' -'other' is not a bufdata class -`byte_size` of other does not match bytesize of self

Each of these conditions will cause similar to return False. Else, returns True.

Used by:

`__eq__ __ne__ bits_different`

...in this class.

`to_buf(myBuf, offset=0)`

Injects information to 'myBuf' starting at 'offset'.

myBuf contains an iterable + indexable object, typically a Twidl buffer, where class information will be inserted.

offset is the location from which the class will start to insert data.

<Historical Note> Parameter order is inverted from old TFW Twidl code; will require refactor as it's implemented. Adoption of PEP 8 naming conventions by this class do the same.

`to_file(file_name, offset=0, append=False)`

Saves information to '*file_name*' in binary format.

Implementation compatible with '`from_file`'.

`txt_to_file(file_name)`

Saves string representation of information to file.

No mirroring '`from txt`', as this would require all classes implement and maintain a text parser; which is non-trivial.

`class src.software.parse.bufdata.BufDataList`

Bases: [`src.software.parse.bufdata.BufData`](#)

Ordered list of BufData that implement the BufData interface.

in combination with 'endian' module; should make 'BufferDataDecode' obsolete; though it is not incompatible. Compatibility is achieved from the common interface, BufData.

`append(member, name=None)`

Adds a member to this ordered list.

If a name is provided, member is made accessible as an attribute.

`append_to_file(file_name, offset=0)`

Saves additional information to 'file_name' in binary format.

Implementation compatible with 'to_file'.

Most bufdata/dataparse implementations won't make much use of this. The only known one so far is burn-in log, which is simply an endless stream of repeating elements: log records.

`base_str()`

over-rides user implemented string functions

`bits_different(other)`

Compares this bufdata structure to 'other'.

Returns the number of bits that differ between the 2.

Raises Exception if similar() is false. There's no good way to report that the comparison could not be made.

Check similar if you're concerned about breaking in this manner

Note that this can be a very time-expensive operation.

`byte_size()`

Reports data size in bytes in buffer.

See bufdata.BufData (super class) for detail.

`file_info()`

Returns a sting containing the file name, created datetime, and file size,

returns " if object never accessed a file.

`from_buf(myBuf, offset=0)`

Extracts listdata from a buffer.

See `bufdata.BufData` (super class) for detail.

`from_file(file_name, offset=0)`

Loads binary information from '*file_name*'.

Implementation compatible with '`to_file`'.

`isAllFFs()`

checks to see if all the data as represented in the buffer is all 0xFF.

Useful, as erased nand is 0xFF

`last_file = None`

`last_file_stat = None`

`name = None`

`similar(other)`

Check if....

-'*other*' is 'None' -'*other*' is not a `bufdata` class -`byte_size` of *other* does not match bytesize of self

Each of these conditions will cause `similar` to return False. Else, returns True.

Used by:

`__eq__` `__ne__` `bits_different`

...in this class.

`to_buf(myBuf, offset=0)`

Inserts list data in to a buffer.

See bufdata.BufData (super class) for detail.

`to_file(file_name, offset=0, append=False)`

Saves information to 'file_name' in binary format.

Implementation compatible with 'from_file'.

`txt_to_file(file_name)`

Saves string representation of information to file.

No mirroring 'from txt', as this would require all classes implement and maintain a text parser; which is non-trivial.

`class src.software.parse.bufdata.DirectAccess`

Bases: [`src.software.parse.bufdata.BufData`](#)

Inheriting class of bufdata that provides a standardized interface to a data member. Currently used to provide BufDataList.member access to BufDataList.

`append_to_file(file_name, offset=0)`

Saves additional information to 'file_name' in binary format.

Implementation compatible with 'to_file'.

Most bufdata/dataparse implementations won't make much use of this. The only known one so far is burn-in log, which is simply an endless stream of repeating elements: log records.

`bits_different(other)`

Compares this bufdata structure to 'other'.

Returns the number of bits that differ between the 2.

Raises Exception if similar() is false. There's no good way to report that the comparison could not be made.

Check similar if you're concerned about breaking in this manner

Note that this can be a very time-expensive operation.

`byte_size()`

Return the number of bytes (int) this uses in a buffer.

<Historical Note> Previous implementation in TFW Twidl set this as an attribute; will require refactor as it's implemented. Adoption of PEP 8 naming conventions by this class do the same.

`file_info()`

Returns a sting containing the file name, created datetime, and file size,

returns " if object never accessed a file.

`from_buf(myBuf, offset=0)`

Extracts information from 'myBuf' starting at 'offset'.

myBuf contains an iterable + indexable object, typically a Twidl buffer, which contains the information to be parsed.

offset is the location from which the class will start to extract data.

It is typical to see this optionally called from `__init__`, especially on classes that extract data that is read to a buffer from drive

<Historical Note> Parameter order is inverted from old TFW Twidl code; will require refactor as it's implemented. Adoption of PEP 8 naming conventions by this class do the same.

`from_file(file_name, offset=0)`

Loads binary information from 'file_name'.

Implementation compatible with 'to_file'.

`get_direct_access()`

Interface to the information stored in an direct access bufdata class.

`isAllFFs()`

checks to see if all the data as represented in the buffer is all 0xFF.

Useful, as erased nand is 0xFF

`last_file = None`

`last_file_stat = None`

`name = None`

`set_direct_access(value)`

Interface to the information stored in an direct access bufdata class.

`similar(other)`

Check if....

-'other' is 'None'
-'other' is not a bufdata class
-byte_size of other does not match bytesize of self

Each of these conditions will cause similar to return False. Else, returns True.

Used by:

`__eq__` `__ne__` `bits_different`

...in this class.

`to_buf(myBuf, offset=0)`

Injects information to 'myBuf' starting at 'offset'.

myBuf contains an iterable + indexable object, typically a Twidl buffer, where class information will be inserted.

offset is the location from which the class will start to insert data.

<Historical Note> Parameter order is inverted from old TFW Twidl code; will require refactor as it's implemented. Adoption of PEP 8 naming conventions by this class do the same.

`to_file(file_name, offset=0, append=False)`

Saves information to 'file_name' in binary format.

Implementation compatible with 'from_file'.

`txt_to_file(file_name)`

Saves string representation of information to file.

No mirroring 'from txt', as this would require all classes implement and maintain a text parser; which is non-trivial.

`class src.software.parse.parserXmlGen.parserXmlGenerator`

Bases: [`src.software.parse.autoObjects.outputGenerationHelper`](#)

`CreateFiles(objList, outputDir=None)`

`capFirstLetter(name)`

`convertToCamelCase(name1, name2)`

`isSpecialName(name)`

`nameSeparator = '_'`

`subdirName = 'xmldata'`

`class src.software.parse.parserXmlGen.xmlFileGenerator(abspath, name='telemetry_csdp')`

Bases: `object`

`addClassDef(structName, structType, fieldList, token=0, majorVersion=0, minorVersion=0)`

`generateFile(telemetryVersionMajor, telemetryVersionMinor)`

Brief:

headerTelemetry.py - Holds special parsers for Telemetry Log Page Header Data

Description:

-

Classes:

Enter GetHeaders("parsersnvmeheaderTelemetry.py") to display Class listings.

class

src.software.parse.headerTelemetry.NvmeControllerInitiatedLogPageHeader_struct

Bases: `_ctypes.Structure`

Brief:

ControllerInitiatedLogPageHeader_struct() - Definition of the NVMe Get Controller Initiated Log Page header structure for Intel drives.

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

IEEE_OUI_id

Structure/Union member

IsHostInitiated()

IsVersion1()

ctrlInitiatedDataAvail

Structure/Union member

ctrlInitiatedGeneration

Structure/Union member

dataArea1EndBlock

Structure/Union member

dataArea2EndBlock

Structure/Union member

dataArea3EndBlock

Structure/Union member

getCIdata()

getDataAreaLastBlock(*dataArea*)

getLastBlock()

getSerialNumber()

getStr()

logIdentifier

Structure/Union member

reasonId

Structure/Union member

reserved

Structure/Union member

reserved2

Structure/Union member

tofile(*filename*)

tostr()

validate()

class

src.software.parse.headerTelemetry.NvmeHostInitiatedLogPageHeader_struct

Bases: `_ctypes.Structure`

Brief:

HostInitiatedLogPageHeader_struct() - Definition of the NVMe Get Host Initiated Log Page header structure for Intel drives.

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

IEEE_OUI_id

Structure/Union member

IsHostInitiated()

IsVersion1()

ctrlInitiatedDataAvail

Structure/Union member

ctrlInitiatedGeneration

Structure/Union member

dataArea1EndBlock

Structure/Union member

dataArea2EndBlock

Structure/Union member

dataArea3EndBlock

Structure/Union member

getCIdata()

getDataAreaLastBlock(*dataArea*)

getLastBlock()

getSerialNumber()

getStr()

getVersionInfo()

logIdentifier

Structure/Union member

reasonId

Structure/Union member

reserved

Structure/Union member

reserved2

Structure/Union member

tofile(*filename*)

tostr()

`validate()`

class
src.software.parse.headerTelemetry.SataControllerInitiatedLogPageHeader_struct

Bases: `_ctypes.Structure`

Brief:

SataControllerInitiatedLogPageHeader_struct() - Definition of the SATA Get Controller Initiated Log Page header structure for Intel drives.

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

IEEE_OUI_id

Structure/Union member

IsHostInitiated()

IsVersion1()

ctrlInitiatedDataAvail

Structure/Union member

ctrlInitiatedGeneration

Structure/Union member

dataArea1EndBlock
Structure/Union member

dataArea2EndBlock
Structure/Union member

dataArea3EndBlock
Structure/Union member

getCIdata()

getDataAreaLastBlock(*dataArea*)

getLastBlock()

getSerialNumber()

getStr()

logIdentifier
Structure/Union member

reasonId
Structure/Union member

reserved
Structure/Union member

reserved2
Structure/Union member

tofile(*filename*)

tostr()

validate()

class
src.software.parse.headerTelemetry.SataHostInitiatedLogPageHeader_struct
Bases: `_ctypes.Structure`

Brief:

SataHostInitiatedLogPageHeader_struct() - Definition of the SATA
Get Host Initiated Log Page header structure for Intel drives.

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

IEEE_OUI_id

Structure/Union member

IsHostInitiated()

IsVersion1()

ctrlInitiatedDataAvail

Structure/Union member

ctrlInitiatedGeneration

Structure/Union member

dataArea1EndBlock

Structure/Union member

dataArea2EndBlock

Structure/Union member

dataArea3EndBlock

Structure/Union member

getCIdata()

getDataAreaLastBlock(*dataArea*)

getLastBlock()

getSerialNumber()

getStr()

logIdentifier

Structure/Union member

reasonId

Structure/Union member

reserved

Structure/Union member

reserved2

Structure/Union member

tofile(*filename*)

tostr()

validate()

class

src.software.parse.headerTelemetry.TelemetryLogPageHeader_union(*imageBuffer, blockSize*)

Bases: `_ctypes.Union`

Brief:

getControllerInitiatedLogPageHeader_union() - Definition of the NVMe Get Controller Initiated Log Page header structure for Intel drives.

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

Bytes

Structure/Union member

NvmeCtrlInitiated

Structure/Union member

NvmeHostInitiated

Structure/Union member

NvmeLogId

Structure/Union member

SataCtrlInitiated

Structure/Union member

SataHostInitiated

Structure/Union member

SataLogId

Structure/Union member

```
getInterfaceLogPageId(sata=False)
getInterfaceTelemetryHeaderStruct()
getNvmeControllerInitiatedStruct()
getNvmeHostInitiatedStruct()
getNvmeLogPageId()
getStr()
parse(outFile=None)
```

Brief:

telemetry_drive.py - Telemetry test drive

Classes:

logDevice

```
class src.software.parse.telemetry_drive.logDevice(testDrive, hiLog=True)
```

Bases: **object**

Telemetry Log device. Abstracts NVMe/SATA differences

NvmeControllerInitiatedLogId = 8

NvmeHostInitiatedLogId = 7

SataControllerInitiatedLogId = 37

SataHostInitiatedLogId = 36

assertDrive()

cmdnsid = 4294967295

generateEvent()

getAsyncManager()

getDrive()
getInterfaceHeader()
getLogNameBase()
getLogPage(*byteCount*, *startOffset*, *create*, *retainAER*)
getLogPage0(*create*, *retainAER*)
getReadBuffer()
isHiLogTest()
queueAsynchronousEventRequest(*callback*)
recoverAssertedDrive()
resetRecover()
setBlock0Size(*byteCount*)
setCiLog()
setCiNotification(*notify*)
setHiLog()
writeBlock(*telemetryDataFile*, *blockSize*, *offset=None*)

class src.software.parse.autoObjects.ctypedef(*name*)
Bases: **object**

 getName()

class src.software.parse.autoObjects.ctypedefList
Bases: **object**

 Typedef list

`addTypedef(identifier, value)`

`cleanList()`

`isTypedef(token)`

`class src.software.parse.autoObjects.dataElement(name, varTypeStr,
docStr=None, dimList=None, bitField=False, structReference=None,
offset=0)`

Bases: `object`

Data object definition

`addDocStr(docStr)`

`getAlignmentSize()`

`getBitCount()`

`getCtypeType()`

`getDescData()`

`getDoc()`

`getName()`

`getOffset()`

`getSize()`

`getSubstruct()`

`getXmlData()`

`isArray()`

`isBitField()`

isBool()
isChar()
isEnum()
isFloat()
isIntegerType()
isPointer()
isQualifier()
isSigned()
isStructOrUnion()
resetStructSize(*size*)
updateOffset(*offset*)
updateToBitField()

vartypeData = <*src.software.parse.varType.ghsCvarType object*>

class src.software.parse.autoObjects.defineList
Bases: **object**

#define object list

addDefine(*identifier, valueStr*)

cleanList()

isDefine(*token*)

class src.software.parse.autoObjects.enumList
Bases: **object**

enum object list

addEnumName(*name*)

cleanList()

isEnum(*token*)

class src.software.parse.autoObjects.enumValueList

Bases: **object**

addEnumEntry(*identifier*, *value=None*)

cleanList()

getEnumEntry(*identifier*)

class src.software.parse.autoObjects.objectDefine(*objName*,
majorVersion=0, *minorVersion=0*, *uid=0*)

Bases: **object**

ctypeAutoGen_Object definition

getCapName()

getDefaultSubstructName(*structType*)

getMajorVersion()

getMinorVersion()

getName()

getObjStruct()

getSize()

getUid()

setMajorVersion(*version*)
setMinorVersion(*version*)
setObjStruct(*structObject*)
setSize(*size*)
setUid(*uid*)

class src.software.parse.autoObjects.outputGenerationHelper
Bases: **object**

capFirstLetter(*name*)
convertToCamelCase(*name1*, *name2*)
isSpecialName(*name*)

src.software.parse.autoObjects.stripCommentBlocking(*docStr*)

class src.software.parse.autoObjects.structDef(*name*, *typeName*,
majorVersion=0, *minorVersion*=0, *docString*=None, *inline*=True,
uid=None)

Bases: **object**

Container class for all cType Structure definition and some metadata

BITSTRUCT_TYPE = 4

OBJECT_TYPE = 3

STRUCT_TYPE = 1

UNION_TYPE = 2

UNKNOWN_TYPE = 0

addDoc(*docstring*)

addMember(*name*, *varTypeStr*, *docString*=*None*, *dimList*=*None*,
bitField=*False*, *subStruct*=*None*)

finalizeStruct()

getDefaultMemberName()

getDefaultSubstructName(*structType*)

getDisplaySize()

getMajorVersion()

getMemberList()

getMinorVersion()

getName()

getSize()

getStructType()

getTwidlType()

getType()

getUid()

isBitFieldType()

isInline()

isPossibleBitStruct(*unionIntSize*)

nameSeparator = '''

setDescription(*description*)

setMajorVersion(*majorVersion*)

setMinorVersion(*minorVersion*)
setName(*name*)
setNoInline()
setUid(*uid*)
updateToBitField()
vartypeData = <*src.software.parse.varType.ghsCvarType object*>

class src.software.parse.autoObjects.structDefList

Bases: **object**

list of structures objects

cleanList()

createStruct(*name, typeName, majorVersion=0, minorVersion=0, docString=None, inline=True*)

debugPrint()

findStruct(*structName*)

removeDuplicateStructures()

updateStuctData()

class src.software.parse.parserTwidlGen.parserTwidlGenerator

Bases: [src.software.parse.autoObjects.outputGenerationHelper](#)

CreateFiles(*objList, outputDir=None*)

addImport(*uidNum, importFileName, importName, twidlType*)

capFirstLetter(*name*)

convertToCamelCase(*name1*, *name2*)

isSpecialName(*name*)

nameSeparator = '_'

subdirName = 'twidlparser'

class src.software.parse.parserTwidlGen.twidlParserFileGenerator(*abspath*,
uid, *twidlDir=None*)

Bases: **object**

addClassDef(*structName*, *structType*, *fieldList*, *size=0*, *majorVersion=0*,
minorVersion=0)

addImport(*importFileName*, *importName*)

generateFile()

class src.software.parse.varType.ghsCvarType

Bases: [src.software.parse.varType.varType](#)

getBaseSize()

getByteStr()

getEnumSize()

getSize(*varTypeStr*)

getVarTypeList()

isBool(*varTypeStr*)

isChar(*varTypeStr*)

isDouble(*varTypeStr*)

isDoubleDouble(*varTypeStr*)

isEnum(*varTypeStr*)
isFloat(*varTypeStr*)
isFloatType(*varTypeStr*)
isInteger(*varTypeStr*)
isIntegerKeyWord(*keyWord*)
isIntegerOrModifierKeyWord(*keyWord*)
isIntegerOrModifierVarType(*varTypeStr*)
isIntegerType(*varTypeStr*)
isLong(*varTypeStr*)
isLongChar(*varTypeStr*)
isLongLong(*varTypeStr*)
isPointer(*varTypeStr*)
isShort(*varTypeStr*)
isShortChar(*varTypeStr*)
isShortShort(*varTypeStr*)
isSigned(*varTypeStr*)
isSignedChar(*varTypeStr*)
isStruct(*varTypeStr*)
isUnion(*varTypeStr*)
isVarTypeToken(*testToken*)

roundupBitCount(*bitCount*)

class src.software.parse.varType.varType(*compileBaseSize*=32,
pointerModifier='*', *typeDict*=None, *boolSize*=None, *enumSize*=None,
charSize=None)

Bases: **object**

getBaseSize()

getByteStr()

getEnumSize()

getSize(*varTypeStr*)

getVarTypeList()

isBool(*varTypeStr*)

isChar(*varTypeStr*)

isDouble(*varTypeStr*)

isDoubleDouble(*varTypeStr*)

isEnum(*varTypeStr*)

isFloat(*varTypeStr*)

isFloatType(*varTypeStr*)

isInteger(*varTypeStr*)

isIntegerKeyWord(*keyWord*)

isIntegerOrModifierKeyWord(*keyWord*)

isIntegerOrModifierVarType(*varTypeStr*)

isIntegerType(*varTypeStr*)
isLong(*varTypeStr*)
isLongChar(*varTypeStr*)
isLongLong(*varTypeStr*)
isPointer(*varTypeStr*)
isShort(*varTypeStr*)
isShortChar(*varTypeStr*)
isShortShort(*varTypeStr*)
isSigned(*varTypeStr*)
isSignedChar(*varTypeStr*)
isStruct(*varTypeStr*)
isUnion(*varTypeStr*)
isVarTypeToken(*testToken*)
roundupBitCount(*bitCount*)

Brief:

intelObjectList.py - Telemetry object unique identification number list

Description:

•

Classes:

Enter GetHeaders("intelObjectList.py") to display Class listings.

class src.software.parse.intelObjectList.ParserObjectMap

Bases: **object**

`CheckObjectHeader(objectEUID, majorVersion, minorVersion,
dataSize, dataAreaNumber, objectOffset)`

Check the read object header against the map expected values

`GetAPLReservedFileName(outFileBaseName='APL',
dataAreaNumber=3)`

Get the APL reserved area file name

`GetEUIDFileName(objectEUID, objectMajor, objectMinor,
outFileBaseName, objectCpuNumber)`

Convert the EUID value

`GetEUIDHumanName(objectEUID, objectMajor, objectMinor)`

Convert the EUID value

`GetEventNumber(objectEUID, mediaBankId)`

`ParseNlogData(nlogParseList, outFileBaseName)`

`ParseObjectData(objectEUID, majorVersion, minorVersion,
inputParseName, coreNumber, eventDumpNumber, outFileBaseName)`

Call the appropriate parser for the object ID

`findEUID(objectEUID, majorVersion, minorVersion)`

`findEUIDParser(objectEUID, majorVersion, minorVersion)`

`isNlogObject(objectEUID)`

`isSpecialId(objectEUID)`

`isValidObject(objectEUID)`

`tostr()`

Brief:

intelVUTelemetry.py - Holds special parsers for NVMe Telemetry Log Page Data

Description:

-

Classes:

Enter GetHeaders("parsersnvmeintelVUTelemetry.py") to display Class listings.

class
src.software.parse.intelVUTelemetry.intelLogObjectInformationV_1_4_struct

Bases: `_ctypes.Structure`

Brief:

`intelLogObjectInformationV_1_4_struct()` - Object entry information version 1.4

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

`getHumanName()`

`objectTextIdentifier`

Structure/Union member

`tostr()`

class
src.software.parse.intelVUTelemetry.intelLogObjectInformation_struct
Bases: `_ctypes.Structure`

Brief:

intelLogObjectInformation_struct() - Object entry information
version 1.

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

getHumanName()

objectEnum

Structure/Union member

objectTextIdentifier

Structure/Union member

tostr()

tstCmdLowId

Structure/Union member

class
src.software.parse.intelVUTelemetry.intelTelemetryDataAreaValidation_uni
on(*imageBuffer*)

Bases: `_ctypes.Union`

Brief:

`intelTelemetryDataAreaValidation_union()` - Data Area Validation Data object.

Description:

This class extends `TWIDL_Union`.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

Bytes

Structure/Union member

`getValidationValue()`

`validate()`

`validation`

Structure/Union member

class

`src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_4_struct`

Bases: `_ctypes.Structure`

Brief:

`intelTelemetryObjectHeader_V1_4_struct()` - Version 1.4 Intel Telemetry data object header

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

cpuNumber

Structure/Union member

flags

Structure/Union member

getCpuNumber()

getDataOffsetAndSize(*offset, size*)

getHumanName()

getMajorVersion()

getMinorVersion()

getObjectByteSize(*tocSize*)

getObjectIdentifier()

objectHeaderSize = 4096

objectIdentifier

Structure/Union member

objectSizeBlks

Structure/Union member

objectTextIdentifier

Structure/Union member

reserved1

Structure/Union member

tostr()

validate()

class

src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_struct

Bases: `_ctypes.Structure`

Brief:

intelTelemetryObjectHeader_V1_struct() - Version 1 Intel Telemetry data object header

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

cpuNumber

Structure/Union member

flags

Structure/Union member

getCpuNumber()

getDataOffsetAndSize(*offset, size*)

getHumanName()

getMajorVersion()

getMinorVersion()

getObjectByteSize(*tocSize*)

getObjectIdentifier()

objectHeaderSize = 4096

objectIdentifier

Structure/Union member

objectSizeBlks

Structure/Union member

objectTextIdentifier

Structure/Union member

reserved1

Structure/Union member

tostr()

validate()

class

src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V2_struct

Bases: `_ctypes.Structure`

Brief:

intelTelemetryObjectHeader_V2_struct() - Version 2 Intel Telemetry data object header

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

getCpuNumber()

getDataOffsetAndSize(*offset, size*)

getHumanName()

getMajorVersion()

getMinorVersion()

getObjectByteSize(*tocSize*)

getObjectIdentifier()

majorVersion

Structure/Union member

mediaId

Structure/Union member

minorVersion

Structure/Union member

objectEUID
Structure/Union member

objectHeaderSize = 12

reserved
Structure/Union member

tostr()

validate()

class
src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_union(*imageBuffer, blockSize*)
Bases: `_ctypes.Union`

Brief:
intelTelemetryObjectHeader_union() - Definition of the NVMe Get Host Initiated Log Page header structure for Intel drives.

Description:
This class extends TWIDL_Union.

Class(es):
None

Method(s):
None

Related: -

Author(s):
Randal Eike

Bytes
Structure/Union member

`getStruct(majorVersion=1, minorVersion=0)`

version1

Structure/Union member

version1_4

Structure/Union member

version2

Structure/Union member

`class src.software.parse.intelVUTelemetry.intelTelemetryTOC_V1_struct`

Bases: `_ctypes.Structure`

Brief:

`intelTelemetryTOC_V1_struct()` - Definition of the Intel Telemetry table of contents version 1.

Description:

Intel VU telemetry data table of contents implementation.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

`VERSION1_TOC_ENTRY_COUNT = 127`

`buildList(dataAreaSize)`

`getDataSize()`

`getEntry(index)`

getEntryCount()
getMajorVersion()
getMinorVersion()
getTOCSize()
majorVersion
 Structure/Union member
minorVersion
 Structure/Union member
objects
 Structure/Union member
reserved
 Structure/Union member
tostr()
validate()

class src.software.parse.intelVUTelemetry.intelTelemetryTOC_V2_struct
Bases: **_ctypes.Structure**

Brief:
intelTelemetryTOC_V2_struct() - Definition of the NVMe Get Host/Controller Initiated Log Page Data area header structure for Intel drives.

Description:
Data area table of contents structure definition

Class(es):
None

Method(s):

None

Related: -

Author(s):

Randal Eike

VERSION2_TOC_ENTRY_COUNT = 200

areaSize

Structure/Union member

buildList(*dataAreaSize*)

entryCount

Structure/Union member

getDataSize()

getEntry(*index*)

getEntryCount()

getMajorVersion()

getMinorVersion()

getTOCSize()

majorVersion

Structure/Union member

minorVersion

Structure/Union member

objects

Structure/Union member

reserved

Structure/Union member

tostr()

validate()

class

src.software.parse.intelVUTelemetry.intelTelemetryTOC_VersionStruct

Bases: `_ctypes.Structure`

Brief:

intelTelemetryTOC_struct() - Definition of the NVMe Get Telemetry Log Page header structure for Intel drives.

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

getMajorVersion()

getMinorVersion()

getStr()

getVersion()

majorVersion

Structure/Union member

`minorVersion`

Structure/Union member

`tostr()`

class

`src.software.parse.intelVUTelemetry.intelTelemetryTOC_union(imageBuffer, blockSize)`

Bases: `_ctypes.Union`

Brief:

`intelTelemetryTOC_union()` - Definition of the NVMe Get Host Initiated Log Page header structure for Intel drives.

Description:

This class extends `TWIDL_Union`.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

Bytes

Structure/Union member

Version

Structure/Union member

`getStruct()`

`v1Struct`

Structure/Union member

v2Struct

Structure/Union member

class
src.software.parse.intelVUTelemetry.telemetryTOCObjectEntryV2_struct
Bases: `_ctypes.Structure`

Brief:

`telemetryTOCObjectEntryV2_struct()` - Definition of the Intel VU telemetry table of contents entry version 2.

Description:

Telemetry object Catalog entry structure definition and method implementation.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

contentSizeBytes

Structure/Union member

`getBlockSize()`

`getByteSize()`

`getDictionaryEntry()`

`getStartingBlock()`

`getStartingOffset()`

`offsetBytes`

Structure/Union member

`tostr()`

`validate()`

class

`src.software.parse.intelVUTelemetry.telemetryTObjectCatalogEntry_struct`

Bases: `_ctypes.Structure`

Brief:

`telemetryTObjectCatalogEntry_struct()` - Definition of the Intel VU telemetry table of contents catalog entry version 1.

Description:

Telemetry object Catalog entry structure definition and method implementation.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

`getBlockSize()`

`getByteSize()`

`getDictionaryEntry()`

`getStartingBlock()`

`getStartingOffset()`

startingBlock
Structure/Union member

tostr()

totalBlocks
Structure/Union member

validate()

Brief:

intelVUTelemetryReason.py - Intel VU reason code structure

Description:

•

Classes:

Enter GetHeaders("intelVUTelemetryReason.py") to display Class listings.

class
src.software.parse.intelVUTelemetryReason.intelTelemetryReasonCode
Bases: `_ctypes.Structure`

Brief:

intelTelemetryReasonCode() - Reason code data

Description:

Class(es):
None

Method(s):
None

Related: -

Author(s):
Randal Eike

controllerInitiated
Structure/Union member

errorDetected
Structure/Union member

excursionDetected
Structure/Union member

getStr()

reasonCode
Structure/Union member

reserved
Structure/Union member

tofile(*filename*)

tostr()

warningDetected
Structure/Union member

class
src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct

Bases: `_ctypes.Structure`

Brief:

`intelTelemetryReasonV1_0_Struct()` - Definition of the NVMe Reason data for intel get telemetry log page commands

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

aplReserved

Structure/Union member

blRevision

Structure/Union member

errorDetected()

excursionDetected()

failureModeString

Structure/Union member

fwRevision

Structure/Union member

getFamilyID()

getMajorVersion()

getMinorVersion()

getReasonCode()

getSerialNumber()

getStr()

getVersion()

intelReserved

Structure/Union member

isControllerInitiated()

majorVersion

Structure/Union member

minorVersion

Structure/Union member

reason

Structure/Union member

serialNumber

Structure/Union member

tofile(*filename*)

tostr()

validate(*hiLog*)

warningDetected()

Brief:

OutputLog.py - Output / display functions

Description:

•

Classes:

output_log

Function(s):

getWindowSize(handle) - Get the selected window size

pressReturnToContinue(spacing = "") - Prompt user to press return,
indent prompt by spacing string

class src.software.parse.output_log.OutputLog

Bases: `object`

Brief:

`output_log()` - Output utility class

Description:

Output utility log functions.

Class(es):

`output_log`

Method(s):

`setDebugLevel(debugLevel = 0)` - Set the debug message output level. All messages with a debug level less than or equal to this level will be displayed.

All messages with a debug level greater than this level will be ignored.

`setWarnIsError(treatWarningsAsError = False)` - If the input is true then all warning messages are sent to the error output stream. If the input is

false then all warning messages are sent to the standard output stream

`Error(errorString)` - Output the input string to the error stream with the ERROR: string prefix

`Warning(warningString)` - Output the input string to the proper stream with the proper WARNING: or ERROR: string prefix. The proper stream is determined

by the state of `OutputLog.warningIsError`

`DebugPrint(debugLevel, informationString)` - If the current value of `OutputLog.debugOutputLevel` is \geq the input `debugLevel` then output the input `informationString`

to the standard output stream. Else do nothing.

`Print(informationString)` - Output the input `informationString` to the standard output stream.

Related:

Author(s):

Randal Eike

`static DebugPrint(debugLevel, informationString)`

`static Error(errorString)`

`static Information(informationString)`

`static Print(informationString)`

`static Warning(warningString)`

`debugOutputLevel = 0`

`static disableQuiet()`

`static enableQuiet()`

`quiet = False`

`static setDebugLevel(debugLevel=0)`

`static setWarnIsError(treatWarningsAsError=False)`

`warningIsError = False`

`src.software.parse.output_log.getWindowSize(handle)`

`src.software.parse.output_log.pressReturnToContinue(spacing='')`

`src.software.parse.output_log.promptUser(prompt)`

This module overlays telemetry binary data over generated data structs and returns a plain text file, or dynamic initialized object python dictionary

containing the objects' metadata and payload

@author: achamorr, jdtarang, reike, ptran, jmadasse, ssekara, dgarces

Args:

--bin: String for the path to the directory containing the bin files to be parsed
--outloc: String for the name of the output file where the Objects' information will be stored as plain text
--projdir: String for the path to the firmware build directory containing the python parsers
--debug: Flag to activate debug statements
--verbose: Flag to activate longer debug statements
--modeSelect: Integer value for run mode (1=bufDict, 2=autoParsers)

Examples:

```
python pacmanIC.py --projdir arbordaleplus_t2 --bin ./sample --debug -  
-verbose --modeSelect 1
```

```
src.software.parse.pacmanIC.main()
```

Parse options and run PacManIC

```
class src.software.parse.pacmanIC.pacManIC(bin_t=None, projDir=None,  
nlogLoc=None, nlogDir=None, outLoc=None, debug=False,  
verbose=False, mode=1)
```

Bases: **object**

overlays telemetry binary data over generated data structs by
ctypeAutoGen

```
pic = pacManIC(bin, projDir, outLoc)  
pic.generateTelemetryObjects() pic.printTelemetryObjects()
```

or to return Dynamic Dictionary of objects:

```
pic.getRunDictionary()
```

Attributes:

binDirectory: String for the path to the directory containing the bin files to be parsed
projectDir: String for the path to the firmware

build directory containing the python parsers
versionedDir: String for the path to the firmware build directory containing the versioned autoParsers
unversionedDir: String for the path to the firmware build directory containing the unversioned autoParsers
outLoc: String for the name of the output file where the Objects' information will be stored as plain text
debug: Boolean flag to activate debug statements
verbose: Boolean flag to activate longer debug statements
mode: Integer value for run mode (1=bufDict, 2=autoParsers)
moduleList: List of python autoParser modules imported so far
listFailedObj: List of object names that failed during parsing
listPassedObjs: List of object names that were successfully parsed
telemetryObjectsGenerated: List of telemetry objects successfully parsed
passedBinFileMetadata: List of bin file metadata for objects that were successfully parsed
failedBinFileMetadata: List of bin file metadata for objects that were not successfully parsed
runDictionary: Dictionary containing the parsed objects metadata and payload

generateTelemetryObjects()

API for parsing the binary files and extracting their payloads into the run dictionary.

Returns:

generateTelemetryObjectsAutoParsers()

function for parsing the binary files and extracting their payloads into the run dictionary. Assumes that pacManIC instance is running in mode 2 and that the firmware build directory has all the parsers contained inside autoParsers/versioned and autoParsers/unversioned

Returns:

generateTelemetryObjectsBuffDict()

function for parsing the binary files and extracting their payloads into the run dictionary. Assumes that pacManIC instance is running in mode 1 and that the firmware build directory has all the parsers contained inside telemetry/parsers

Returns:

`getDictionary()`

API to create dictionary containing the parsed objects metadata and payload

Returns:

Dictionary containing the parsed objects metadata and payload

`getEntry(uid, major, minor, core)`

function for getting an entry in the dictionary for objects' metadata and payload

Returns:

Entry in the dictionary for objects' metadata and payload

`getListFailedObjects()`

function for getting the list of object names that failed during parsing

Returns:

List of object names that failed during parsing

`getListPassedObjects()`

function for getting the list of object names that were parsed successfully

Returns:

List of object names that were parsed successfully

`getModuleList()`

function for getting the list of python autoParser modules imported so far

Returns:

List of python autoParser modules imported so far

getRunDictionary()

function for getting the dictionary with objects' metadata and payload

Returns:

Dictionary with objects' metadata and payload

getTelemetryObjects()

function for getting the list of telemetry objects successfully parsed

Returns:

List of telemetry objects successfully parsed

pacmanICAPI()

API to replace standard command line call (the pacManIC class has to instantiated before calling this method)

Returns:

printTelemetryObjects()

API for printing the telemetry objects and their payloads as plain text.

Returns:

printTelemetryObjectsAutoParsers(*headerUID*: int = 240)

function for printing the telemetry objects and their payloads as plain text. It assumes that pacManIC instance is running in mode 2.

Returns:

printTelemetryObjectsBuffDict()

function for printing the telemetry objects and their payloads as plain text. It assumes that pacManIC instance is running in mode 1.

Returns:

`setBinDirectory(bin_t=None)`

function for checking and setting the path to the directory containing the bin files to be parsed

Args:

`bin_t`: String for the path to the directory containing the bin files to be parsed

Returns:

`setProjDirectory(projdir=None)`

function for checking and setting the path to the firmware build directory containing the python parsers

Args:

`projdir`: String for the path to the firmware build directory containing the python parsers

Returns:

`class src.software.parse.parserSrcUtil.fileTokenizer(operatorList=None, delimiterList=None, stringOpList=None, slCommentStartList=None, mlCommentStartList=None, mlCommentEnd=None, continuationCharacter='\\', stripWhite=True)`

Bases: [`src.software.parse.parserSrcUtil.tokenList`](#)

Tokenize structure definition file on a character-by-character level

`DELIMITER_TOKEN = 0`

`EOL_TOKEN = 7`

`IDENTIFIER_TOKEN = 2`

`ML_COMMENT_TOKEN = 3`

`OPERATOR_TOKEN = 1`

`SL_COMMENT_TOKEN = 4`

STRING_TOKEN = 5

WHITESPACE_TOKEN = 6

addToken(*tokenType, tokenValue*)

clear()

getNextToken()

Read the next token from the list of tokens generated by the input stream

getPreviewToken()

Read the next token from the list of tokens generated by the input stream without moving index.

isEndofList()

Check if we have reached the end of the token list

parseStream(*inputStream*)

Generate the token list from the input file

putToken()

Adjust the token list pointer back one token

resetTokenPull()

resetTokenizer()

class src.software.parse.parserSrcUtil.numericParser

Bases: **object**

Parse and validate numeric token

BinDigitList = ['0', '1']

DigitList = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']

HexDigitList = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f', 'A', 'B', 'C', 'D', 'E', 'F']

OctalDigitList = ['0', '1', '2', '3', '4', '5', '6', '7']

getValue(*token*)

isNumber(*token*)

class src.software.parse.parserSrcUtil.parserHelper(*tokenizer*,
keywordList=None, *preprocessorKey=None*)

Bases: [src.software.parse.parserSrcUtil.numericParser](#)

File parser helper

BinDigitList = ['0', '1']

COMMENT_TOKEN = 5

DEFAULT_TOKEN = 0

DELIMITER_TOKEN = 8

DigitList = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']

END_OF_LIST = 9

HexDigitList = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f', 'A', 'B', 'C', 'D', 'E', 'F']

IDENTIFIER_TOKEN = 1

KEYWORD_TOKEN = 2

NUMERIC_TOKEN = 6

OPERATOR_TOKEN = 3

OctalDigitList = ['0', '1', '2', '3', '4', '5', '6', '7']

PREPROCESSOR_TOKEN = 4

STRING_TOKEN = 7
continueParse()
createNode()
getIgnoreToken()
getNextNode()
getNextToken()
getPreviewToken()
getValue(*token*)
isNumber(*token*)
isValidIdentifier()
parseError(*formatStr*, *args=None*)
parseInformation(*formatStr*, *args=None*)
parseWarning(*formatStr*, *args=None*)
putToken()
resetParser()

class src.software.parse.parserSrcUtil.tokenList

Bases: **object**

Token list handler

DELIMITER_TOKEN = 0

EOL_TOKEN = 7

IDENTIFIER_TOKEN = 2

ML_COMMENT_TOKEN = 3
OPERATOR_TOKEN = 1
SL_COMMENT_TOKEN = 4
STRING_TOKEN = 5
WHITESPACE_TOKEN = 6

`addToken(tokenType, tokenValue)`

`clear()`

`getNextToken()`
Read the next token from the list of tokens generated by the input stream

`getPreviewToken()`
Read the next token from the list of tokens generated by the input stream without moving index.

`isEndofList()`
Check if we have reached the end of the token list

`putToken()`
Adjust the token list pointer back one token

`resetTokenPull()`

class
`src.software.parse.telemetryCmd.TelemetryObjectCommands(devObj=None)`

Bases: **object**

`parseTelemetry(objectId, inFile=None, projObjFile=None)
read the Telemetry object`

class src.software.parse.structdefParser.structdefParser

Bases: [src.software.parse.parserSrcUtil.parserHelper](#)

Parse structure definition file

BinDigitList = ['0', '1']

COMMENT_TOKEN = 5

CleanList()

DEFAULT_TOKEN = 0

DELIMITER_TOKEN = 8

DigitList = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']

END_OF_LIST = 9

GetObjectList()

GetStructList()

HexDigitList = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f', 'A', 'B', 'C', 'D', 'E', 'F']

IDENTIFIER_TOKEN = I

KEYWORD_TOKEN = 2

KeyWordList = ['struct', 'union', 'ObjectBegin', 'ObjectEnd', '_LINES', 'const', 'static', 'volatile']

NUMERIC_TOKEN = 6

OPERATOR_TOKEN = 3

OctalDigitList = ['0', '1', '2', '3', '4', '5', '6', '7']

OperatorList = ['=', '<', '>', '+', '-', '/', '%', '*', '^', '&', '|', '~', '!', ',', '.', '::']

PREPROCESSOR_TOKEN = 4

STRING_TOKEN = 7
continueParse()
createNode()
defaultNameCounter = 0
findStruct(*structName*)
getIgnoreToken()
getNextNode()
getNextToken()
getPreviewToken()
getValue(*token*)
isNumber(*token*)
isValidIdentifier()
objectList = []
parseDefFile(*inputStream*)
parseError(*formatStr*, args=None)
parseInformation(*formatStr*, args=None)
parseWarning(*formatStr*, args=None)
putToken()
resetParser()
structLevel = 0
structList = <*src.software.parse.autoObjects.structDefList object*>

class src.software.parse.structdefParser.structdefTokenizer
Bases: [src.software.parse.parserSrcUtil.fileTokenizer](#)

Tokenize structure definition file

DELIMITER_TOKEN = 0

EOL_TOKEN = 7

IDENTIFIER_TOKEN = 2

ML_COMMENT_TOKEN = 3

OPERATOR_TOKEN = 1

SL_COMMENT_TOKEN = 4

STRING_TOKEN = 5

WHITESPACE_TOKEN = 6

addToken(tokenType, tokenValue)

clear()

getNextToken()

Read the next token from the list of tokens generated by the input stream

getPreviewToken()

Read the next token from the list of tokens generated by the input stream without moving index.

isEndofList()

Check if we have reached the end of the token list

parseStream(inputStream)

Generate the token list from the input file

putToken()

 Adjust the token list pointer back one token

resetTokenPull()

resetTokenizer()

class src.software.parse.parserDictionaryGen.parserDictionaryGenerator
Bases: [src.software.parse.autoObjects.outputGenerationHelper](#)

CreateFiles(*objList*, *outputDir=None*)

capFirstLetter(*name*)

convertToCamelCase(*name1*, *name2*)

isSpecialName(*name*)

nameSeparator = '_'

subdirName = 'tokenparser'

src.software.parse.internal.twidlDictGen.DATACONTROLCSV =
'/tools/lib/datacontrol/structures.csv'

THIS SCRIPT ASSUMES USE OF ARBORDALEPLUS_CA BUILD
FOR STRUCTURE EXTRACTION, EDIT FOR VARIABILITY
LATER NOTE: THIS RUNS ON NAND ONLY CURRENTLY

class src.software.parse.internal.twidlDictGen.OpenFiles

Bases: **object**

Managing files files = OpenFiles()

use open method foo = files.open("text.txt", "r")

close all files files.close()

close()

`open(*args)`

`src.software.parse.internal.twidlDictGen.getDictionary(uidList=None)`

`src.software.parse.internal.twidlDictGen.getOverlappingSet(str1, str2)`

`src.software.parse.internal.twidlDictGen.getTelemetryItemsFromFileNamesList(fileList)`

takes as input a text file with names of telemetry binaries

`src.software.parse.internal.twidlDictGen.runCmd(multicmd)`

Performs an execution of GHS program.

`src.software.parse.internal.twidlDictGen.test()`

`src.software.parse.internal.twidlDictGen.testSize()`

`class src.software.parse.internal.twidlDictGen.twidlDictGenerator`

Bases: `object`

`auditAvailableStructures()`

`@todo: Should audit which structure can run ctypeAutoGen successfully (can generate from c to python ctypes)`

`createDictFromParserObjects()`

`getDictionary()`

`getUid(uidName)`

Map struct Name to appropriate Number uid

`getUidT(uidName)`

Map struct Name to appropriate Number uid, return int uid if possible, None if None, and string uid if not convertible to int

`identifyLatestRunDestDirectory()`

nameToUidConverterGen(*dataControlDict*)

printDict()

runCtypeAutoGenScript()

Brief:

drive_utility.py - Test drive selection and identification utility for generic tests

Description:

-

Classes:

testDrive, driveList

Function(s):

ScanDrives(option,opt_str,value,parser), SetUlink(ulinkCommand)

Related:

-

Author(s):

Randal Eike, Joe Tarango

src.software.parse.internal.drive_utility.ScanDrives(*option, opt_str, value, parser*)

src.software.parse.internal.drive_utility.SetUlink(*ulinkCommand*)

class src.software.parse.internal.drive_utility.driveList

Bases: **object**

Brief:

driveList() - Utility functions to get a list of potential test drives in the system.

Description:

Class to abstract test drive list functions.

Class(es):

testDrive

Method(s):

`__init__(self)` - Constructor
`getDrive(self, drvIndex = None)` - Scan the system for the requested drive

Related: -

Author(s):

Randal Eike

`static checkDriveIndex(driveNumber=None)`

`static displayDriveList()`

`static getDriveFromList(prompt)`

`static reattachDrive(serialNumber, excludeSystemDrive=True)`

`static validDrvIndexList(excludeSystemDrive=True)`

`class src.software.parse.internal.drive_utility.testDrive(drvIndex=None)`

Bases: `object`

Brief:

`testDrive()` - Test drive selection and identification utility class for python test scripts.

Description:

This class implements the generic utility functions required to select and identify a drive to be used to run a generic test script.

Class(es):

testDrive

Method(s):

`__init__(self)` - Constructor
`getDrive(self, drvIndex = None)` - Scan the system for the requested drive

Related: -

Author(s):
Randal Eike

`getFamily()`
`getSerialNumber()`
`getTestDrive()`
`globalDriveSpecificParams()`
`isDriveAsserted()`
`printBasicInfo()`
`toStr()`
`unlockDrive()`

Brief:

`pssDebugTraceTriage.py` - Parse the pssdebug trace list to look for state sequence errors

Description:
PSS Debug state change triage analysis object

Classes:
`pssDebugTraceTriage`.

`src.software.parse.nlogParser.nlog_triage.pssDebugTraceTriage.ABRUPT_SHUTDOWN = 101`

The event transition map defines the PCIE, PCICFG and NVME state transitions associated with the event in the FW

`src.software.parse.nlogParser.nlog_triage.pssDebugTraceTriage.TOOL_VERSION = 0.7`

State lists define the enumeration value associated with the state, the previous allowed states and the name of the state

class

`src.software.parse.nlogParser.nlog_triage.pssDebugTraceTriage.pssDebugTraceTriage(family='CDR')`

Bases:

[src.software.parse.nlogParser.nlog_triage.nlogTriageBase.nlogTriageBase](#)

Process the input Debug state and look for unexpected or illegal transitions

`addNlogEvent(formatStr, params, tsSeconds, coreId)`

Determine if this a pss debug trace nlog entry and record the state transition in the list if it is

`@param formatStr - nlog format string @param params - nlog associated parameter tuple @param tsSeconds - nlog timestamp in seconds (floating point value) @param coreId - Core number for the NLOG`

`checkEvents()`

Check the nlog tuple list for illegal PSS state transitions

`@return list of strings for issues found`

`errorLevel = 0`

`eventList = []`

`getVersion()`

`information = 2`

`updateTime(seconds)`

warningLevel = 1

Brief:

padrTriage.py - Parse the PADR nlog message

Description:

Triage the PADR messages and look for errors

Classes:

padrTriage.

class src.software.parse.nlogParser.nlog_triage.padrTriage.padrTriage

Bases:

[src.software.parse.nlogParser.nlog_triage.nlogTriageBase.nlogTriageBase](#)

addNlogEvent(format, params, tsSeconds, coreId)

Determine if this a pss debug trace nlog entry and record the state transition in the list if it is

@param format - nlog format string @param params - nlog associated parameter tuple @param tsSeconds - nlog timestamp in seconds (floating point value) @param coreId - Core number for the NLOG

checkEvents()

Check the nlog tuple list for illegal PSS state transitions

@return list of strings for issues found

errorLevel = 0

getVersion()

information = 2

padrActionStartMarker = '_'

padrActionStartMarkerLen = 2

```
padrMarker = 'PADR_'
padrMarkerLen = 5
padrStateEndMarker = '_'
transitionMarker = '-> goto '
transitionMarkerLen = 9
updateTime(seconds)
warningLevel = 1
```

Brief:

nlogTriageBase.py - Base object for nlog triage

Description:

Base class for nlog triage objects

Classes:

pssDebugTraceTriage.

class

src.software.parse.nlogParser.nlog_triage.nlogTriageBase.nlogTriageBase(*t
riageName, TOOL_VERSION*)

Bases: **object**

Nlog triage base object

addNlogEvent(formatStr, params, tsSeconds, coreId)

Determine if this a pss debug trace nlog entry and record the state transition in the list if it is

@param formatStr - nlog format string @param params - nlog associated parameter tuple @param tsSeconds - nlog timestamp in seconds (floating point value) @param coreId - Core number for the NLOG

checkEvents()

Check the nlog tuple list for illegal PSS state transitions

@return list of strings for issues found

errorLevel = 0

getVersion()

information = 2

updateTime(*seconds*)

warningLevel = 1

Brief:

hostTimeMarkerTriage.py - Parse the host time marker message

Description:

Host timestamp marker triage

Classes:

HostTimeMarkerTriage.

class

src.software.parse.nlogParser.nlog_triage.hostTimeMarkerTriage.HostTimeMarkerTriage

Bases:

[src.software.parse.nlogParser.nlog_triage.nlogTriageBase.nlogTriageBase](#)

static ConvertToWallClock(*utcHostTimeMs*, *utcHostTimeSec*,
powerOnTimeMs)

static GetNlogTimestamp(*tsSeconds*, *flag*)

addNlogEvent(*formatStr*, *params*, *tsSeconds*, *coreId*)

Determine if this a pss debug trace nlog entry and record the state transition in the list if it is

@param formatStr - nlog format string @param params - nlog associated parameter tuple @param tsSeconds - nlog timestamp in seconds (floating point value) @param coreId - Core number for the NLOG

checkEvents()

Check the nlog tuple list for illegal PSS state transitions

@return list of strings for issues found

errorLevel = 0

getVersion()

information = 2

localTimeZone = *None*

updateTime(*seconds*)

warningLevel = 1

Brief:

Level2Parser.py - Level 2 parser utility

Description:

Execute the level 2 parser

Classes:

telemetryHostTime.

class

src.software.parse.nlogParser.telemetry_parsers.Level2Parser.Level2Parser(
objectParserName, *structSize*=0)

Bases: **object**

`parseData(binFileName, outputTextName=None)`

Parse the data object and output the text

`@param binFileName - Path/Name of the binary file with the data to parse` `@param outputTextName - Path/Name of the output text file or None to use binfileName.txt`

`@retval - Bool True = parsed without error, False = parse failed`

class

`src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.EventHeader_struct`

Bases: `_ctypes.Structure`

Brief:

`EventHeader_struct()` - Data structure definition for Event Header

Description:

This class is the structure of Event Header data read from the drive using Direct Access TC @ base address 0xFFFF50000.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Phuong P Tran

EventNum

Structure/Union member

NumParams

Structure/Union member

SourceNum

Structure/Union member

class
src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.EventHeader_union(*dwordList, currentOffset*)

Bases: `_ctypes.Union`

Brief:

EventHeader_union() - Union structure definition to load the Event Header structure (event ID) with the data from the nlog dword list

Description:

Event ID definition (first 1 word) of the nlog entry.

Class(es):

EventHeader_struct

Method(s):

`__init__(dwordList, currentOffset)` object getStruct()

Related:

EventTimeStamp_struct, EventId_struct

Author(s):

Phuong P Tran, modified Randal Eike to include timestamp

EMPTY_TOKEN = 0

HOST_TIME_SET_TOKEN = 4294966790

MAX_PARAMETER_COUNT = 8

SOURCE0_EVENT_LIST = [2690]

TIME_ADJUSTMENT_TOKEN = 4294967042

WALL_CLOCK_EPOCH_TOKEN = 4294966785

dword

Structure/Union member

getDwordCount()

getEventNum()

getFormatDictionaryKey()

getNumParams()

getSourceNum()

header

Structure/Union member

isEmpty()

isHostTimeSetEvent()

isTimeAdjustEvent()

isValid()

isWallClockEpochTimeEvent()

class

src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.EventTime
Stamp_struct

Bases: `_ctypes.Structure`

Brief:

`EventTimeStamp_struct()` - Data structure definition for Event Header timestamp

Description:

This class is the timestamp nlog event structure.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

getTime()

seconds

Structure/Union member

ticks

Structure/Union member

time0Marker

Structure/Union member

class

src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.EventTime
Stamp_union(*dwordList, currentOffset*)

Bases: `_ctypes.Union`

Brief:

EventTimeStamp_union() - Union structure definition to load the Event timestamp structure with the data from the nlog dword list

Description:

Event definition and timestamp (first 3 words) of the nlog entry.

Class(es):

EventId_struct,EventTimeStamp_struct

Method(s):

`__init__(dwordList, currentOffset)` object getStruct()

Related:

EventTimeStamp_struct, EventId_struct

Author(s):

Phuong P Tran, modified Randal Eike to include timestamp

dword

Structure/Union member

static getDwordCount()

getStruct()

Get the entry header structure object

@return object - Entry header data

timeStamp

Structure/Union member

class

src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.EventTuple
Translate(*counterFreq=200000000.0, formatsFile=None,*
enumParserFile=None, inlineTriage=False)

Bases: **object**

Brief:

EventTupleTranslate() - Translate an input event tuple list into a list of strings

Description:

Using the specified nlog_format.py translate a list of nlog event tuples into formatted nlog text strings.

Class(es):

nlogEnumTranslate

Method(s):

__init__(counterFreq, formatsFile) string list xlateEvents()

Related:

EventTupleTranslate - Used to generate the event tuple list that this class will translate
OutputLog - Error, debug message handler

Author(s):

Randal Eike, Methods taken from nlogpost2.py and modified to use telemetry tools structures

GetTriageText()

Get the triage message list and header generated by the registered nlog triage objects

RegisterTriageList(*nlogTriageList*)

Register an nlog triage object

@param nlogTriageList - Nlog triage object to register

RegisterTriageObject(*nlogTriageObj*)

Register an nlog triage object

@param nlogTriageObj - Nlog triage object to register

generateHeader(*counterFreq*, *coreId=None*)

markFirstEntry(*nlogTextEvents*)

xlateEvents(*events*)

Translate the given NLog events tuples into a list of strings, formatting each as required.

Based on nlogpost2.py printEvents

@param events - Chronologically ordered list of nlog event tuples
@param coreId - Core number associated with the log

@return - list of strings, list of nlog output strings

src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NLogPoolArray(*data*)

class
src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogEventPoolParser(*nlogName*, *coreId=None*)

Bases: **object**

Brief:

NlogEventPoolParser - Class to aid in the conversion of an nlog dword pool into a list of nlog event tuples

Class(es):

EventId_struct, EventTimeStamp_struct

Method(s):

__init__(*nlogName*) tuple list getEventTupleList(*dwordList*)

Related:

EventTimeStamp_struct, EventId_struct

Author(s):

Randal Eike

getEventTupleList(*dwordList*)

Read and translate the input nlog bin file into an ordered event tuple list. Modified from nlogpost2.py extractEvents().

@param *dwordList* - Dword list of nlog entries or None if we should use the constructor

@return list - Chronologically ordered list of nlog events tuples (zone number 0 | 1, time stamp MSW, time stamp LSW, eventHeader ID structure, param tuple, nlog name string)

setLocalTimeZone(*timeZone*)

Set the timezone according to the input local timezone specification from the log generator.

If this function is not called or if the python version does not support the tzset then calander clock output remains set to UTC time.

@param timeZone - Timezone environment variable

class
src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelect
V2_struct
Bases: `_ctypes.Structure`

Brief:

NlogSelectV2_struct() - Data structure definition for V2.0 TVE
Nlog Select read and write test command

Description:

This is the structure class of the TVE Nlog select read/write test command. Data read and write from the drive using TC -51.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Phuong P Tran

LogSelect

Structure/Union member

Reserved

Structure/Union member

Version

Structure/Union member

coreCount

Structure/Union member

coreSelected

Structure/Union member

getCoreSelected()

getLogByteSize()

getLogId()

getLogName()

getTicksPerSecond()

nlogByteSize

Structure/Union member

nlogId

Structure/Union member

nlogName

Structure/Union member

nlogPauseStatus

Structure/Union member

nlogPrimaryBufferSize

Structure/Union member

ticksPerSecond

Structure/Union member

totalNlogs

Structure/Union member

class
src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelect
V4_struct

Bases: `_ctypes.Structure`

Brief:

NlogSelectV4_struct() - Data structure definition for V4.0 TVE
Nlog Select read and write test command

Description:

This is the structure class of the TVE Nlog select read/write test command. Data read and write from the drive using TC -51.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Phuong P Tran

LogSelect

Structure/Union member

Reserved

Structure/Union member

Version

Structure/Union member

coreCount

Structure/Union member

coreSelected

Structure/Union member

getCoreSelected()

getLogByteSize()

getLogId()

getLogName()

getTicksPerSecond()

nlogBufNum

Structure/Union member

nlogBufNumMax

Structure/Union member

nlogByteSize

Structure/Union member

nlogId

Structure/Union member

nlogName

Structure/Union member

nlogPauseStatus

Structure/Union member

nlogPrimaryBufferSize

Structure/Union member

selectAddedOffset

Structure/Union member

selectNlogPause

Structure/Union member

selectOffsetRef

Structure/Union member

ticksPerSecond

Structure/Union member

totalNlogs

Structure/Union member

class

src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelect
Version_struct

Bases: `_ctypes.Structure`

Brief:

NlogSelectVersion_struct() - Data structure definition for TVE Nlog
Select version identification

Description:

This is the structure class of the TVE Nlog select read/write test
command. Data read and write from the drive using TC -51.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Phuong P Tran

VersionMajor

Structure/Union member

VersionMinor

Structure/Union member

class
src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelect
_struct
Bases: `_ctypes.Structure`

Brief:

NlogSelect_struct() - Data structure definition for TVE Nlog Select read and write test command

Description:

This is the structure class of the TVE Nlog select read/write test command. Data read and write from the drive using TC -51.

Class(es):

None

Method(s):

None

Related: -

Author(s):

Phuong P Tran

LogSelect

Structure/Union member

Reserved

Structure/Union member

TimestampEnd

Structure/Union member

TimestampStart

Structure/Union member

Version

Structure/Union member

getCoreSelected()

getLogByteSize()

getLogId()

getLogName()

getTicksPerSecond()

class

src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelect_union(*imageBuffer, blockSize*)

Bases:

[src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.UnionBase](#)

Brief:

NlogSelect_union() - Top level data structure for TVE Nlog select read/write

Description:

This class is the structure of TVE Nlog select data read from or write to the drive using TC -51.

Bytes

Structure/Union member

Version

Structure/Union member

Version1

Structure/Union member

Version2

Structure/Union member

Version4

Structure/Union member

getStruct()

getStructSize()

parse()

class

src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.TelemetryV
2NlogEventParserL2(*nlogFileName=None, alignment=1*)

Bases: **object**

Brief:

Telemetry nlog level 2 bin file parser. This class will parse the input bin file and generate a list of event tuples

Class(es):

EventId_struct, EventTimeStamp_struct

Method(s):

__init__(*nlogFileName*) tuple list getEventTupleList() int
getCoreId() int getNlogId() string getNlogName() int
getFrequency()

Related:

EventTimeStamp_struct, EventId_struct

Author(s):

Randal Eike

ALIGNED_4K = 3

ALIGNED_512 = 2

PACKED_DATA = 1

`WriteNlogStream(outputStream, headerText=None,
nlogTextList=None)`

Output Standard header, Nlog header and nlog text to the specified output stream

@param *outputStream* - Output stream object to write the data to
@param *headerText* - Nlog Header text or None to skip header output
@param *nlogTextList* - Nlog output string list or None

`WriteTriageStream(outputStream, triageTextList=None)`

Output Standard header, Nlog header and nlog text to the specified output stream

@param *outputStream* - Output stream object to write the data to
@param *triageTextList* - Triage output string list or None

`XlateAndOutputMultipleNlog(nlogFileNameList=None,
formatsFile=None, outputStream=None, triageStream=None)`

Parse the nlog file and output the data

@param *nlogFileNameList* - List of Path/File names of telemetry Nlog objects to parse and translate @param *formatsFile* - Path/File name of the Nlog_formats.py file to use for format translation
@param *outputStream* - Stream object to output translated text to
@param *triageStream* - Stream object to output trage data or None to throw it away

@return bool - True = parse and translation worked. False = error

`XlateAndOutputNlog(formatsFile=None, outputStream=None,
nlogFileName=None, triageStream=None)`

Parse the nlog file and output the data

@param *formatsFile* - Path/File name of the Nlog_formats.py file to use for format translation @param *outputStream* - Stream object to output translated text to @param *nlogFileName* - Path/File name of

the telemetry Nlog object to parse and translate @param
triageStream - Stream object to output trage data or None to throw it away

@return bool - True = parse and translation worked. False = error

getEventTupleList()

Read and translate the input nlog bin file into an ordered event tuple list. Modified from nlogpost2.py extractEvents().

@return list - Chronologically ordered list of nlog events tuples
(zone number 0 | 1, time stamp MSW, time stamp LSW,
eventHeader ID structure, param tuple, nlog name string)

setNlogBinFileName(*nlogFileName*)

xlateToText(*eventList*: Optional[list] = None, *nlogFormats*=None, *enumFile*=None)

class

src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.UnionBase(
imageBuffer, *blockSize*)

Bases: `_ctypes.Union`

parse()

class

src.software.parse.nlogParser.telemetry_parsers.nlogEnum.nlogEnumTranslate(*enumParserFile*=None)

Bases: `object`

Brief:

`nlogEnumTranslate()` - Translate the format and parameters into enum name strings

Description:

Using the specified nlogEnumParser.py translate the format string into enum names as appropriate.

Class(es):

OutputLog

Method(s):

`__init__(enumParserFile = None, inlineTriage = False)`
`newFormatString, newParamsTuple enumParser(format, params)`

Related:

Author(s):

Matt Zweig - Methods taken from nlogpost2.py
Randal Eike -
Modified to use telemetry tools structures and added enum format id processing

`enumFormatIdEndMarker = ')'`

`enumFormatIdEndMarkerLen = 1`

`enumFormatIdMarker = '%('`

`enumFormatIdMarkerLen = 2`

`enumParser(format, params)`

Add your nlog format to be parsed below. Make sure that when you add the enum-dict translation above, you create a the key as a decimal integer, and the value as a string. If you don't you'll get type errors

`@param format - Format string for the nlog message` `@param params - Message parameter value tuple`

`@return string - String with enum values inserted`

Brief:

`telemetryHostTime.py` - Current timestamp from the device

Description:

Timestamp structure from the device. Timestamp could be UTC time from the host + power on time or just power on time

Classes:

telemetryHostTime.

class

src.software.parse.nlogParser.telemetry_parsers.telemetryHostTimeV1_0.ho
stTimeFlags

Bases: `_ctypes.Structure`

Brief:

hostTimeFlags() - Flags data used to determine how to interpret the timestamp value

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

isHostTimeStamp()

loadData(*dataBuffer*, *offset*=0)

origin

Structure/Union member

reserved

Structure/Union member

class
src.software.parse.nlogParser.telemetry_parsers.telemetryHostTimeV1_0.telemetryHostTimeV1_0

Bases: `_ctypes.Structure`

Brief:

`telemetryHostTimeV1_0()` - Telemetry host timestamp data object parser

Description:

Class(es):

None

Method(s):

None

Related: -

Author(s):

Randal Eike

flags

Structure/Union member

`loadData(dataBuffer, offset=0)`

timestamp

Structure/Union member

`tofile(fileName=None)`

Brief:

`output_log.py` - Output / display functions

Description:

•

Classes:

`test_util.output_log`

Function(s):

`getWindowSize(handle)` - Get the selected window size

`pressReturnToContinue(spacing = "")` - Prompt user to press return, indent prompt by spacing string

`class src.software.parse.nlogParser.test_util.output_log.OutputLog`

Bases: `object`

Brief:

`test_util.output_log()` - Output utility class

Description:

Output utility log functions.

Class(es):

`test_util.output_log`

Method(s):

`setDebugLevel(debugLevel = 0)` - Set the debug message output level. All messages with a debug level less than or equal to this level will be displayed.

All messages with a debug level greater than this level will be ignored.

`setWarnIsError(treatWarningsAsError = False)` - If the input is true then all warning messages are sent to the error output stream. If the input is

false then all warning messages are sent to the standard output stream

`Error(errorString)` - Output the input string to the error stream with the ERROR: string prefix

`Warning(warningString)` - Output the input string to the proper stream with the proper WARNING: or ERROR: string prefix. The proper stream is determined

by the state of OutputLog.warningIsError

DebugPrint(debugLevel, informationString) - If the current value of OutputLog.debugOutputLevel is \geq the input debugLevel then output the input informationString
to the standard output stream. Else do nothing.

Print(informationString) - Output the input informationString to the standard output stream.

Related:

Author(s):

Randal Eike

static DebugPrint(debugLevel, informationString)

static Error(errorString)

static Information(informationString)

static Print(informationString)

static Warning(warningString)

debugOutputLevel = 0

static disableQuiet()

static enableQuiet()

static enableSilentMode()

static nlogFormatError(errorType, formatStr, params)

Output nlog format error message to the error stream

@param errorType - Error type string @param formatStr - format string in error @param params - parameter tuple

quiet = *False*
static restoreMode(*oldDebugLevel*)
static setDebugLevel(*debugLevel*=0)
static setWarnIsError(*treatWarningsAsError*=*False*)
silent = *False*
warningIsError = *False*

src.software.parse.nlogParser.test_util.output_log.getWindowSize(*handle*)
src.software.parse.nlogParser.test_util.output_log.pressReturnToContinue(*s*
pacing=")
src.software.parse.nlogParser.test_util.output_log.promptUser(*prompt*)

Rapid Automated-Analysis for Developers (RAAD) Framework Sample

The art of Automating Systematic Analysis through Telemetry Meta-Data.

Summary

Telemetry is the state space snapshot which tightly-couple specialists to pertinent data, remotely, removing the cyber physical challenges with interacting on complex platforms. The immediate benefit is precise and rapid data extraction correlated to customer platforms. The purposeful subsequent benefit is reactive-proactive real-time analytics for monitoring of client platforms and data centers. The real-time processing of the data enables data mining, machine learning, and artificial intelligence. The application of these techniques is given in an instance of Intel SSDs and can be applied to technological eco-systems.

Citing

Bibtex

```
@misc{TarangoIntelRAAD2022,
author = {Joseph Tarango, Intel, et al.},
title = {Rapid Automated-Analysis for Developers (RAAD) by Intel},
year = {2022},
publisher = {GitHub Intel},
journal = {GitHub repository Intel},
howpublished = {\url{http://github.com/intel/raad}},
commit = {main}
}
```

If you use the source code, please give credit to the authors.

- Title of program/source code
 - Rapid Automated-Analysis for Developers (RAAD)
- Date
 - 02-26-2018
- Code version
 - 1.0, Pre-Alpha
- Type
 - Source Code
- Sponsor(s)
 - Intel Corporation
 - Non-volatile Memory Storage Group (NSG)
 - Labs
 - Frank Hady
 - Fellow of AI Systems Memory/Storage at Intel
 - <https://www.linkedin.com/in/frankhady/>
 - Pradeep Dubey
 - Senior Fellow of Intel Labs at Intel
 - <https://www.linkedin.com/in/pradeep-dubey-a5592a53/>
- Primary Investigator, Architect, Designer, Director, and Lead Developer
 - Joseph (Joe) David Tarango
 - Machine Learning Engineer of Non-volatile Memory Storage Group (NSG) at Intel
 - <http://josephtarango.com>
 - <https://www.cs.ucr.edu/~jtarango/>
 - <https://www.github.com/jtarango>
 - <https://www.linkedin.com/in/joseph-tarango-451695a2>
- Control Flag Architect, Artificial Intelligence Researcher of Intel

- Niranjan Hasabnis
 - <https://www.linkedin.com/in/niranjan-hasabnis-62a6a33>
- Code Simularity Architect, Artifical Intelligence Researcher of Intel
 - Shengtian Zhou
 - <https://www.linkedin.com/in/shengtian-zhou>
- Business Unit Sponsor and Liaison
 - Jim Baca
 - (Former) Principal Engineer of Non-volatile Memory Storage Group (NSG) at Intel
 - <https://www.linkedin.com/in/jim-baca-657747>
- Research Professor, University of California at Riverside
 - Philip Brisk
 - <https://profiles.ucr.edu/app/home/profile/pbrisk>
- Internship Engineers Focused on RAAD:
 - Daniel Garces
 - Harvard Ph.D. Candidate
 - <https://www.linkedin.com/in/daniel-garcесb/>
 - Tyler Woods
 - University of California at Riverside, Master's Candidate
 - <https://www.linkedin.com/in/tyler-woods-112354172/>
 - Rogelio Macedo
 - University of California at Riverside, Master's Candidate
 - <https://www.linkedin.com/in/rmace001/>
 - Andrea Chamorro
 - Colorado of University at Boulder Bachelor's Candidate
 - <https://www.linkedin.com/in/andrea-chamorroq/>

- Sungkeun Kim
 - Texas A&M Ph.D. Candidate
 - <https://www.linkedin.com/in/sungkeun-kim-20b898117/>
- Zijia (Stella) Cao
 - University of Chicago Master's Candidate
 - <https://www.linkedin.com/in/stella-zjcao/>
- Special Thanks to former Collaborator, Machine Programming Research Director
 - Justin Gottschlich
 - (Former) Principal Artificial Intelligence Scientist of Intel Labs at Intel
 - <http://justingottschlich.com>
- Acknowledgement(s), Contributor(s), and Champion(s)
 - Abdullah Mueen
 - <https://www.cs.unm.edu/~mueen/>
 - Abdullah Muzahid
 - <http://people.tamu.edu/~abdullah.muzahid/index.html>
 - Anand Venkat
 - <https://www.linkedin.com/in/anvenkat>
 - Andrew (Andy) Sainz
 - <https://www.linkedin.com/in/andrew-sainz-b925a993>
 - Bradley (Brad) MacDonald
 - <https://www.linkedin.com/in/brad-m>
 - Chetan Kumar Gupta
 - Chin-Chia Michael Yeh
 - <https://www.cs.ucr.edu/~myeh003/>
 - David Escamilla

- <https://www.linkedin.com/in/david-escamilla-709b193>
- Emmon Keogh:
 - <https://www.cs.ucr.edu/~eamonn/>
- Fangke Yeh
 - <https://scholar.google.com/citations?user=6dp6cJ4AAAAJ>
- Javier Turek
 - <https://www.intel.com/content/www/us/en/research/researchers/javier-turek.html>
- Jean Mary Madassery
 - <https://www.linkedin.com/in/jean-mary-madassery-46461464>
- Jesmin Jahan Tithi
 - <https://www.intel.com/content/www/us/en/research/researchers/jesmin-jahan-tithi.html>
- Jordan Howes
 - <https://www.linkedin.com/in/jordan-howes-59360111>
- Kaveh Kamgar:
 - <https://www.linkedin.com/in/kaveh-kamgar>
- Krishnamurthy Viswanathan:
 - <https://www.linkedin.com/in/krishnamurthy-viswanathan-14b65a42>
- Lukasz Tur
 - <https://pl.linkedin.com/in/lukasz-tur-9574094>
- Mejbah Alam:
 - <https://www.intel.com/content/www/us/en/artificial-intelligence/bios/mohammad-mejbah-ul-alam.html>

- Michael Yeh
- Nesime Tatbul
 - <https://people.csail.mit.edu/tatbul/>
- Pallavi Dhumal
- Paul Peterson:
 - <https://www.linkedin.com/in/paul-petersen-41355011>
- Phoung Tran
- Randal (Randy) Eike
 - <https://www.linkedin.com/in/randal-eike-124931167>
- Ryan Marcus
 - <https://www.csail.mit.edu/person/ryan-marcus>
- Shengtian Zhou
 - <https://www.linkedin.com/in/shengtian-zhou>
- Subhashini Sekaran
- Tim Kraska
 - <https://people.csail.mit.edu/kraska/>
- Timothy Mattson
 - <https://www.intel.com/content/www/us/en/research/researchers/tim-mattson.html>
- Vivek Sarkar
- Yan Zhu
- Zachery Zimmerman:
 - <https://www.linkedin.com/in/zpzim>
- University of California at Riverside (UCR)
 - <https://www.ucr.edu/>

Developer Information

Hello Fellow Developers,

The repository is currently in **pre-alpha**; which means there are many bugs, mixing of language sets (I.E. Python 2.x/3.x. and C/C++ standards), and unfinished code sets. The code is being released for reference and not all members may be participating. Please feel free to fix the code through pull requests.

- Please note for the best experience use **Ubuntu 22.04 LTS x86_64** (<https://releases.ubuntu.com/22.04/>) as the development environment.
- To use a virtual machine or fast path Ubuntu install then navigate to the link below
 - https://github.com/intel/RAAD/blob/main/vm_ubuntu_22.04_x86_64/Readme.md
- New development should have a unit test capability built in to ensure there are no regressions.
 - Auto doxygen location

RAAD\dox\build\index.html

- Docstring Style
 - https://sphinxcontrib-napoleon.readthedocs.io/en/latest/example_google.html
- Example documentation execute order

```
cd RAAD\dox\source\  
python findClasses.py  
cd ..  
make clean  
make html
```

- The main documentation page is at the link below
 - <https://github.com/intel/RAAD/blob/main/dox/source/RAAD.rst>
- An example python template of a can be seen in

RAAD\src\software\utility\templateUtility.py

Organization

- .raadProfile (hidden folder) Folder containing user preference and cache files.
- _dev_tools (Folder) Developer assistant source code.
- data (folder) Reference information for unit tests.
- dox (folder) Code auto documentation generator.
 - build (folder) Active documentation build.
 - doc (folder) Contains various documentation with RAAD.
 - architecture (folder) Template of creating an architecture feature.
 - images (folder) Supporting diagrams.
 - interns (folder) Intern exit presentations.
 - papers (folder) White papers of RAAD.
 - presentations (folder) Research, development, and product instrumentation.
 - source (folder) source code to build documentation.
- scripts (folder) Binary build, installer, and helping scripts.
- src (Folder) Source Code.
 - mpr (Folder) Source Code for machine programming.
 - autoperf (Folder) External Library for performance analysis on code.
 - controlflag (Folder) External Library for software anomaly analysis.
 - mism (Folder) External Library for code similarity.
 - software (Folder) Software Source.
 - access (Folder) modules to access telemetry information.
 - autoAI (Folder) modules to construct and run neural nets dynamically in a multi-layer model.
 - axon (Folder) modules to upload telemetry content into a webpage tool used in searching of traits.
 - container (Folder) modules to dynamically manage data objects.

- cpt (Folder) modules to dynamically classify and predict data properties.
- dAMP (Folder) modules for domain automated machine programming.
- datacontrol (Folder) Data control assistance modules.
- decode (Folder) Decoder modules.
 - ADP_UV (Folder) Ctype auto parser generated decoder modules (not public).
 - CDR_DA (Folder) Ctype auto parser generated decoder modules (not public).
- DP (Folder) modules for data preprocessing
- external (Folder) External tools for workload generation.
- JIRA (Folder) modules to use Atlassian database.
- MEP (Folder) modules for media error prediction using the auto aggressive moving average model variants.
- mp (Folder) modules for time series analysis through Matrix Profile.
- parse (Folder) modules to decode a storage data container.
- probeTrace (Folder) modules to interact with a Green Hill Debug Probe.
- sourceManagement (Folder) source code interface.
- TSV (Folder) modules to generate and visualize time series.
- twidl (Folder) external library to import helper modules (not public).
- utility (Folder) module template for new development.
- vm_ubuntu_22.04_x86_64 Ubuntu 22.04 LTS virtual machine install scripts and image zipped

Related Repositories

1. <https://github.com/IntelLabs/control-flag.git>
2. <https://github.com/IntelLabs/MICSAS.git>
3. <https://github.com/jtarango/CPUBenchmark.git>
4. <https://github.com/mejbah/AutoPerf>

License

Copyright (C) <2019-2022> Intel Corporation

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the
License.

You may obtain a copy of the License at

<https://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing,
software distributed under the License is distributed on an "AS
IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.

See the License for the specific language governing permissions
and limitations under the License.

SPDX-License-Identifier: Apache-2.0

Apache License Version 2.0, January 2004
<http://www.apache.org/licenses/LICENSE-2.0>

Abstract

Within industry, novel endeavors typically spans across disciplines, domains, and parallel activation paths merging modules to bypass previous technological bottlenecks and limitations. The fusion of these technologies introduce intermodal interfaces, yielding convoluted state dependency chains and time series activation paths. For a user single request, our modern systems can decompose and generate large sets of micro operations yielding large amounts of meta data known as telemetry. Historically, our engineers could manage telemetry these data sets for a kilo bytes of telemetry data and as we approach giga bit; it has become an unwieldy big data challenge.

To ease the decomposition and analysis of our big data, our teams brainstormed requirements to automation data management within the source code so our experts can focus Technological and Strategic Long Range Planning (TSLRP). As a consequence of collaboration, we developed ubiquitous standards to identify/extract firmware, hardware, software, and system aggregations in developer's native programming languages to minimize the disruption product pipelines. Using the native languages as input sources, we utilized compiler intermediate representations to transform across languages such that we could use object-oriented inheritance to construct a unsupervised labeling, relational, and data structure management model we call "data control".

Our data control model enables, stochastic (machine learning) and deterministic (formal) methods. Stochastic methods include machine learning, such as deep neural networks, reinforcement learning, genetic algorithms, Bayesian networks, and others. Deterministic methods include formal methods, such as formal verifiers, spatial and temporal logics, formal program synthesizers, and more. The fusion of these method is known as "Machine Programming" and is targeted to facilitate organizational scaling. Our Telemetry and Machine Programming fusion has yielded a tool set known as Rapid Automated Analysis for Developers (RAAD). RAAD's venture is foremost to advance technical collaboration

with our customers in an unprecedented manner, and is key in enabling the next generation of client devices and data centers.

Rapid Automation and Analysis for Developers (RAAD) using Telemetry 2.0 User Guide

J.Tarango, Et al., Application Number, Patent Title :

- 20210191726, Methods And Apparatus For Continuous Monitoring Of Telemetry In The Field
- 20210073632, Methods, Systems, Articles Of Manufacture, And Apparatus To Generate Code Semantics
- 20210157512, Flexible Configuration Of Storage Device With Translation Language Set
- 20200379687, Storage Device With Client Reconfigurable Protocol
- 20190042129, Technologies For Adjusting The Performance Of Data Storage Devices Based On Telemetry Data
- 20200226080, Solid State Drive With External Software Execution To Effect Internal Solid-state Drive Operations
- Additional pending...

Introduction

Title of Project: Pioneering Rapid Fault Analysis using Telemetry for Enhanced Customer Experience (R-FATE) for Solid State Drives

Summary

Telemetry is the state space snapshot which tightly-couples specialists to pertinent data, remotely, removing the cyber physical challenges with interacting on complex platforms. The immediate benefit is precise and rapid data extraction correlated to customer platforms. The purposeful subsequent benefit is proactive real-time analytics for monitoring of client platforms and data centers.

The topology to notify, extract, verify, tokenize, consolidate, and analyze data structures through an automated approach shipping on our products today. The real-time processing of the data enables data mining, machine learning, and artificial intelligence exploration.

In an adaptable commodity market, Intel's® platform facilitates diligence across preceding barriers to accelerate innovation and time-to-market in an unprecedented tactical approach. The innovation within the storage interfaces from Intel's® customer driven focus has catapulted an umbrella of methods and apparatuses focus on rapid development, platform stability, and collaborative early adoption of our state-of-the-art products. The initiatives have driven the prolific timeframes for Patent to Product seen in NSG's history related to Solid-State-Drive (SSD) telemetry.

Opportunity/Problem

The cyber-physical, engineer resourcing, and availability limits the performing state to an average of 3 weeks for triage of the platform. The best known methods (BKM) for driving resolution requires Application Engineers (AEs) on site with a high probability of the device returned to Intel's internal development teams. Furthermore, the BKM entails disjoint procedures for acquiring applicable logs, dumps, traces, etc. prone to human error not verifiable on site. The consequence associated to these events is large risk exposure time frames of fault, product delays, and affirmation dissociation of the Intel® Brand by customers.

When investigating the retrospectives of challenges, we can confirmed common basis is time-to-data (TTD). TTD is defined by determinate timeline from the fault appearance to the confirmation of adequate evidence to proceed a technical resolution strategy. Thus, TTD is the optimizable variable within the telemetry initiative. Telemetry is designed to be Intel's remote method for a device to capture and save pertinent data. In general, telemetry is to be the BKM for fault analysis of anomaly events internally in ConVal, PV and externally in customer engagements. Telemetry does not do any analysis. The key ease of use is we no longer need to unlock the device since the access method is available through standard commands.

Within Intel, we have streamlined through automation for the following: pulling, extracting, and parsing.

Solution

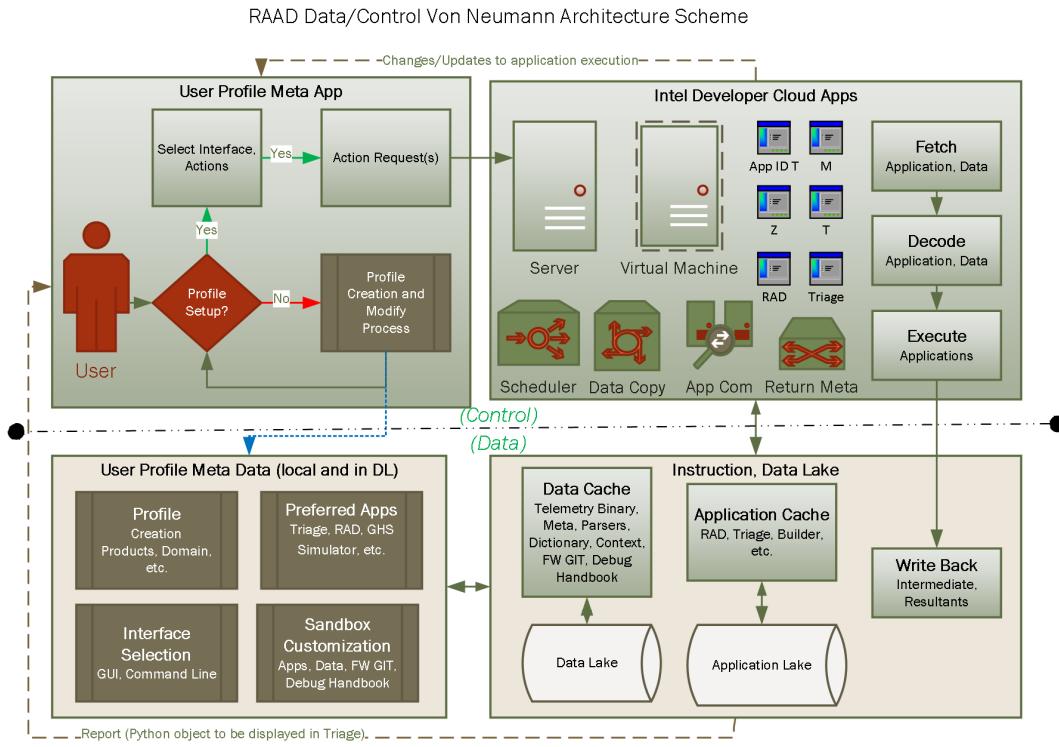
The introduction of Intel telemetry enables the Meta data to be remotely captured and relayed to Intel through a standardized interface. Our proprietary intelligent instantiation ensures all pertinent Meta data is tailored for the signature observable by Telemetry. The topology reduces the customer engagement and data confirmation from a few weeks down to minutes. The advantage of our feature is it captures an internal state snapshot of previously volatile data at the time of failure. The locality and persistence of Meta data to the time of fault increases the probability of key operational behaviors correlated to the module, software, firmware, and/or component introducing the anomaly.

Capabilities

RADD's approach is a scalable framework infrastructure to run workload, capture, extract, parse content, construct a unique profile signature, generate developer assistance graphs, cluster previous signatures in database tracking systems, similarity measure it to previous signatures, speculative root cause based on developer wiki pages, meta statistical/artificial intelligence model forecasting for incomplete meta, and generate comprehensive report.

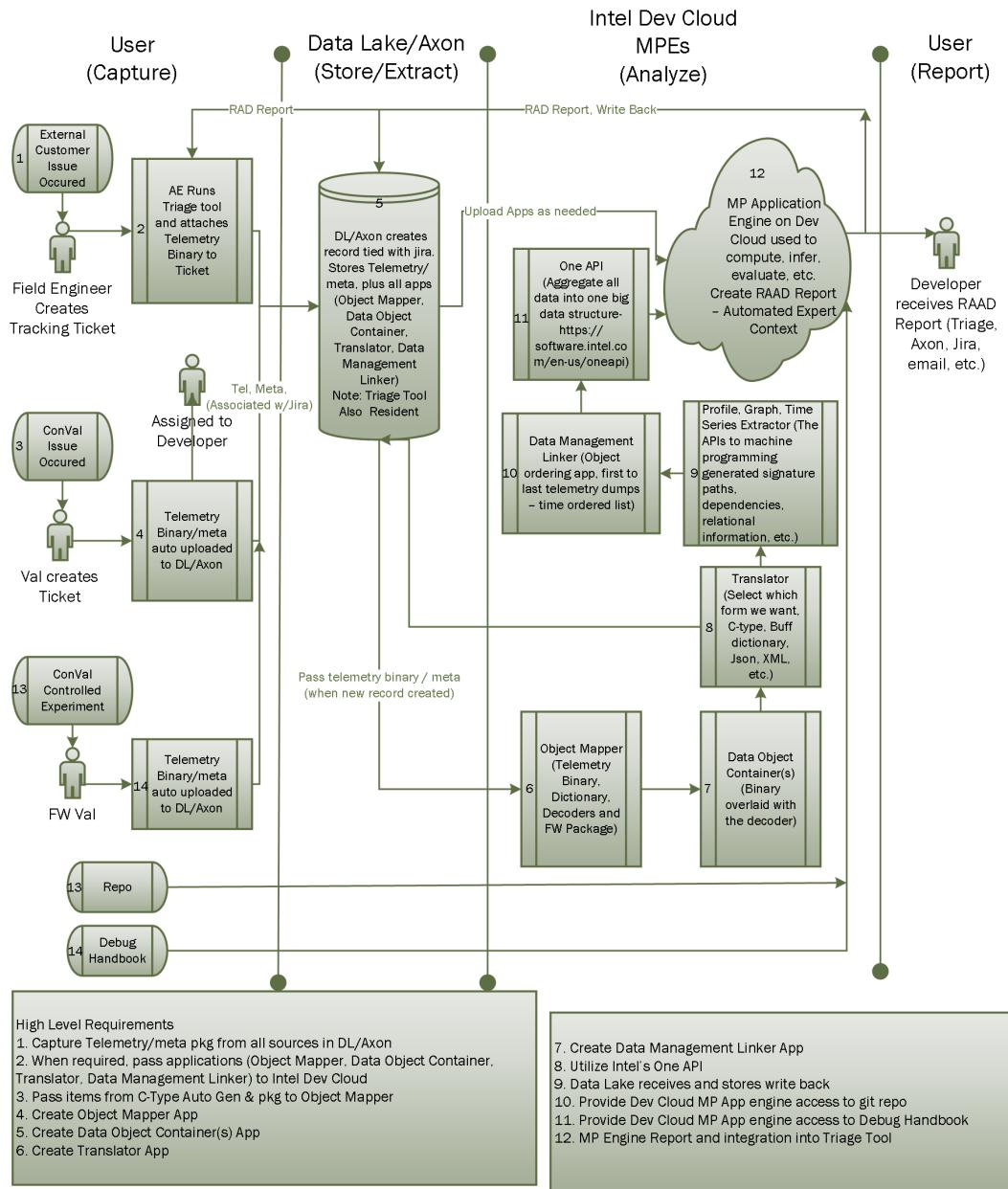
Diagrams

Architecture



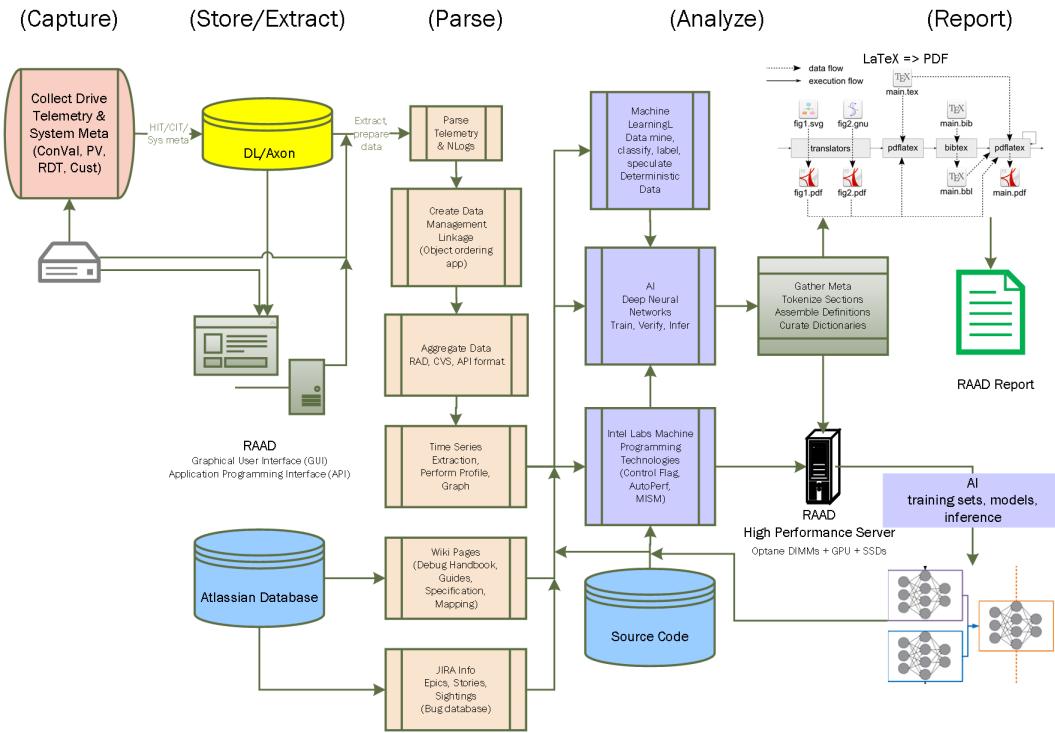
Flow

Data Lake RAAD API (Expert Context Engine)

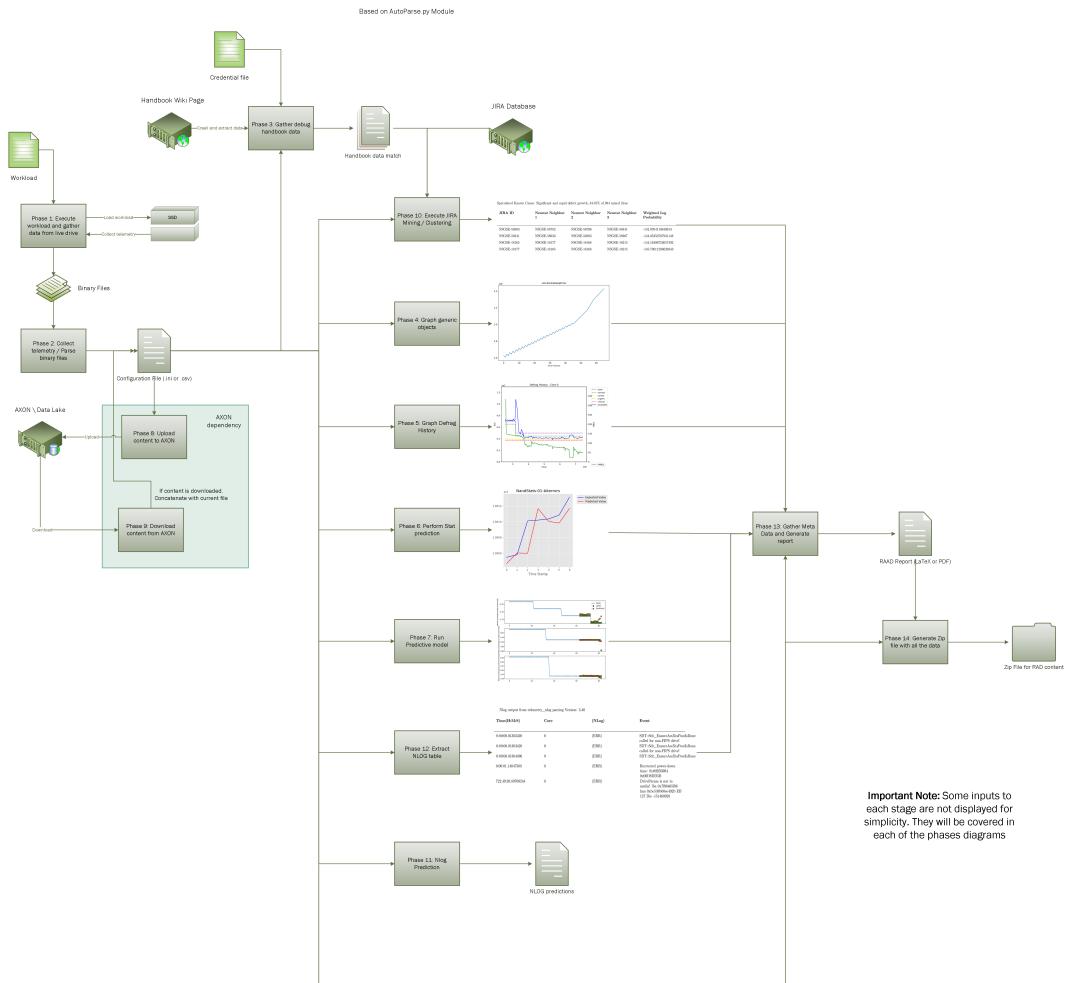


Data Flow

RAAD Data Flow (Expert Context Report)

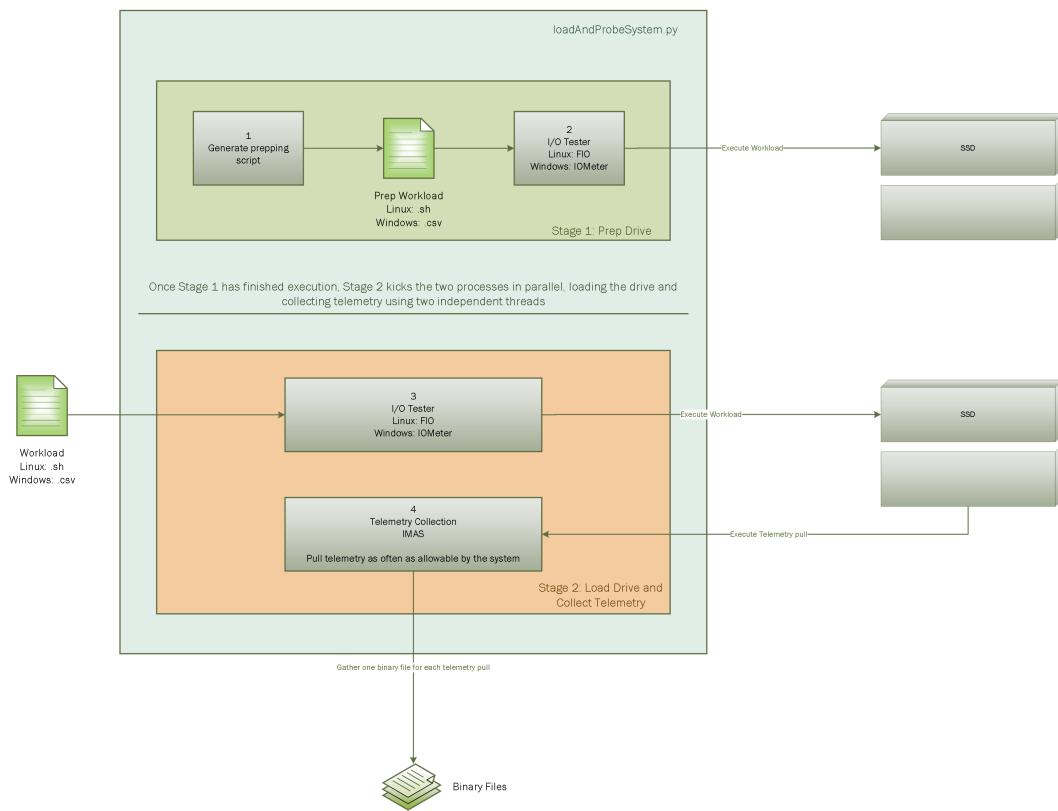


Overview

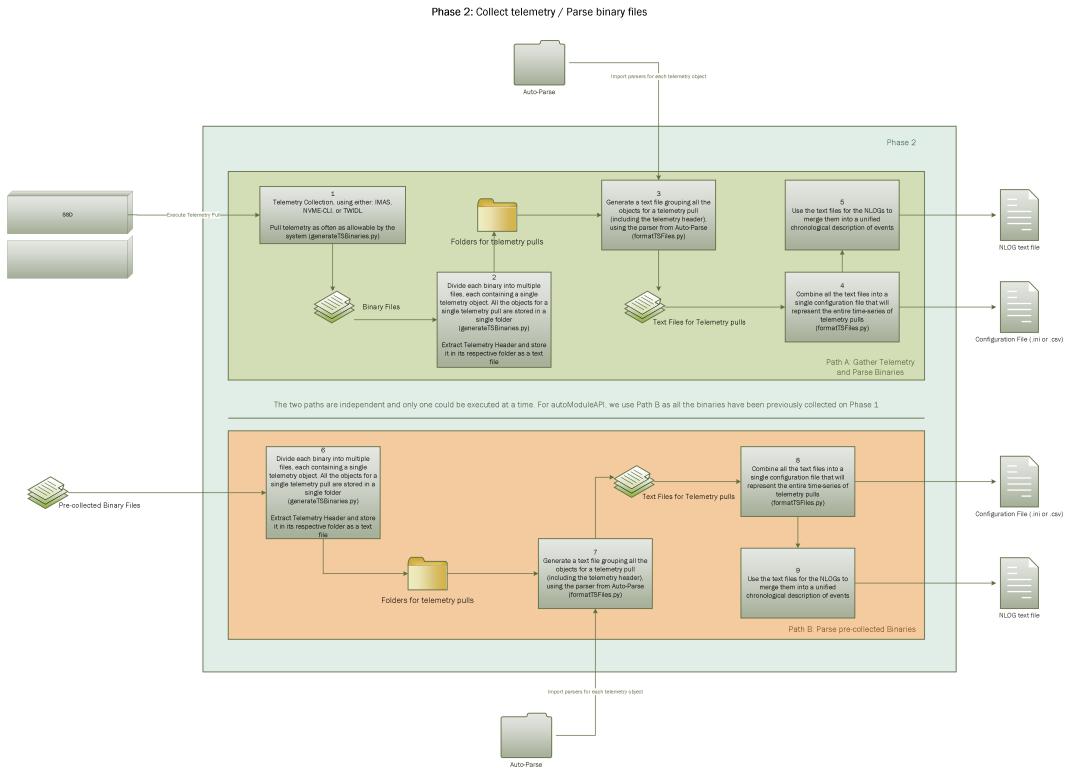


Phase 1

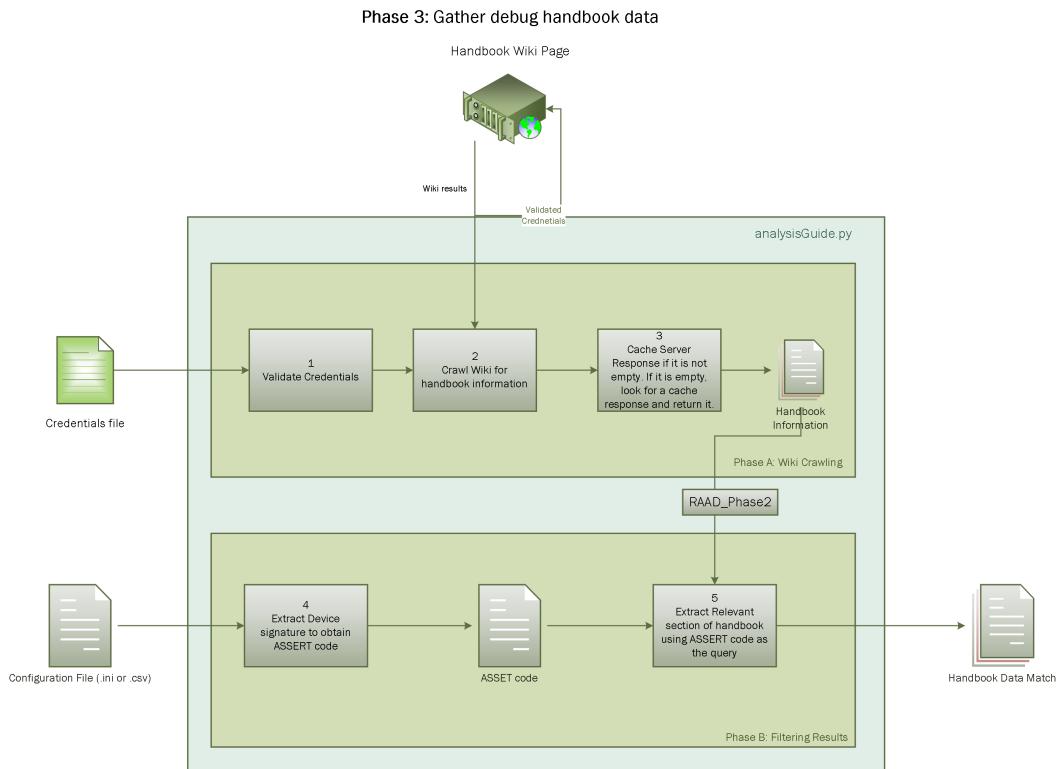
Phase 1: Execute workload and gather data from live drive



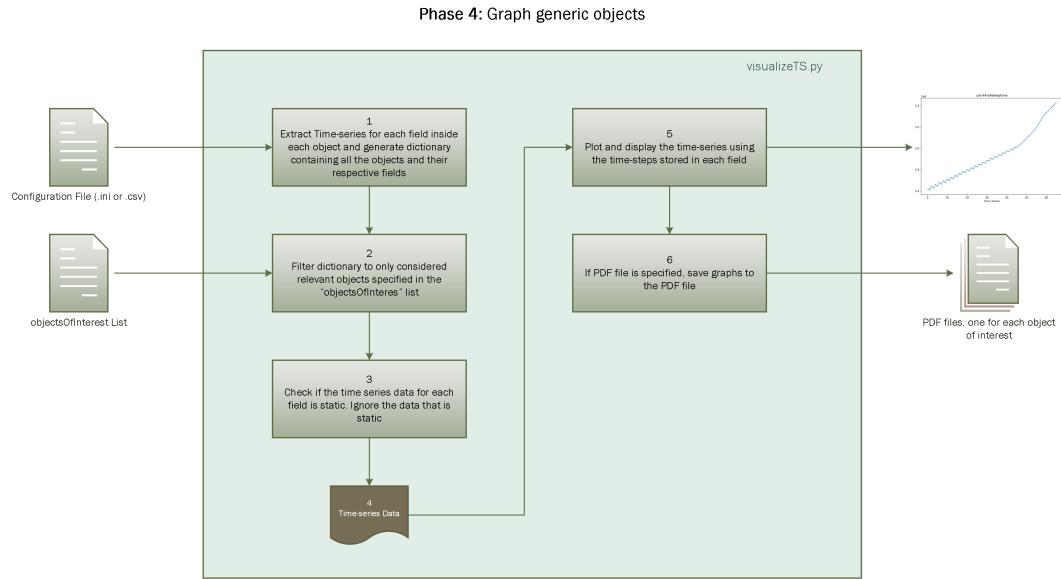
Phase 2



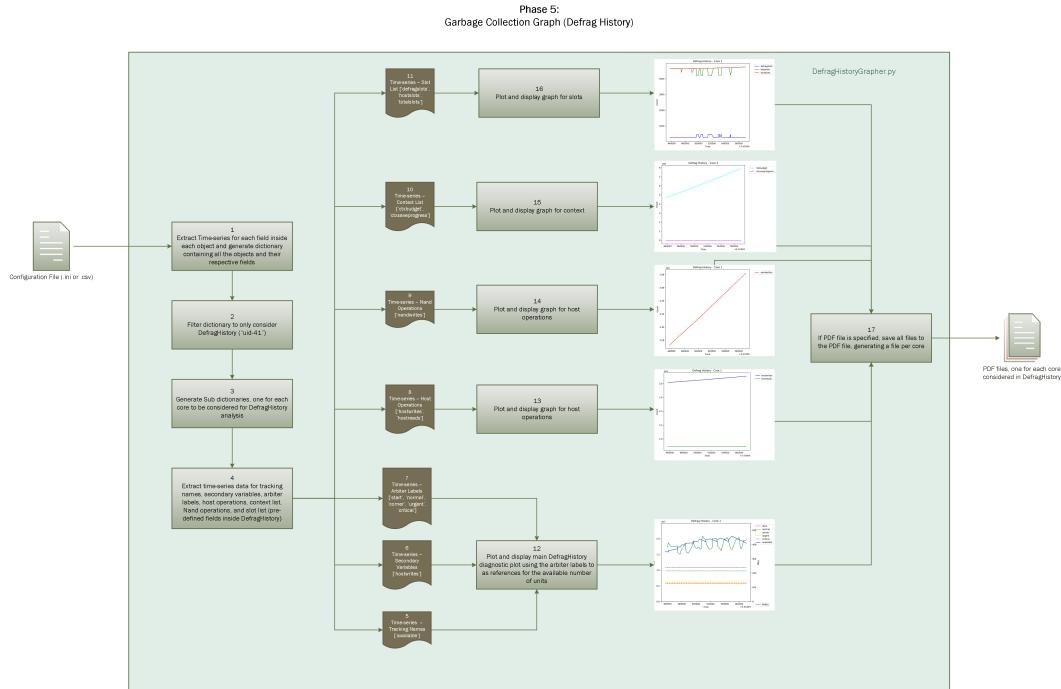
Phase 3



Phase 4

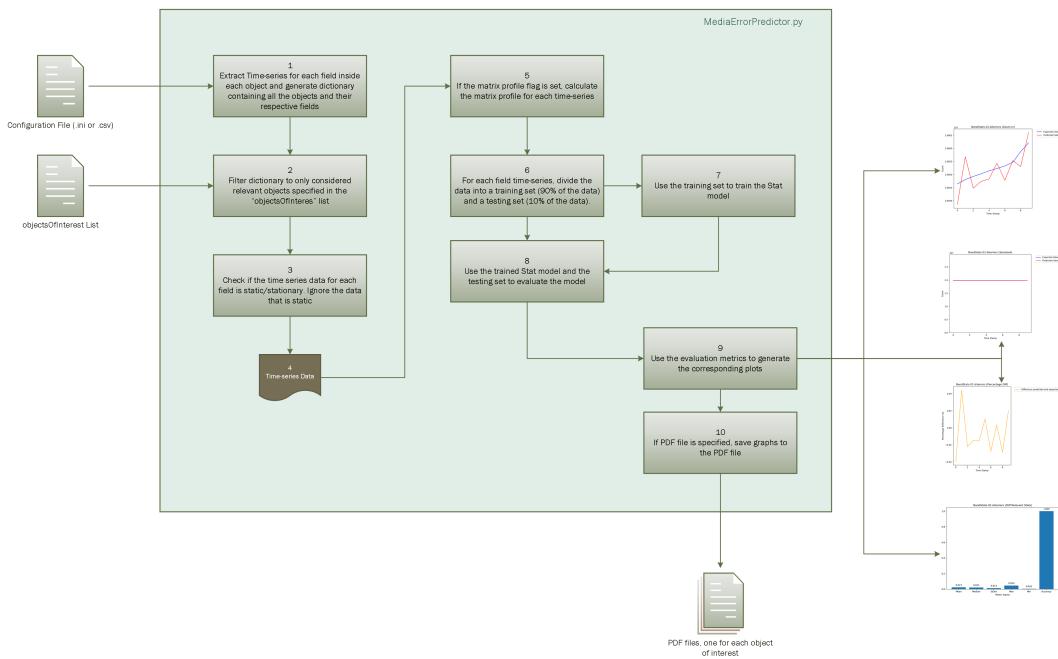


Phase 5



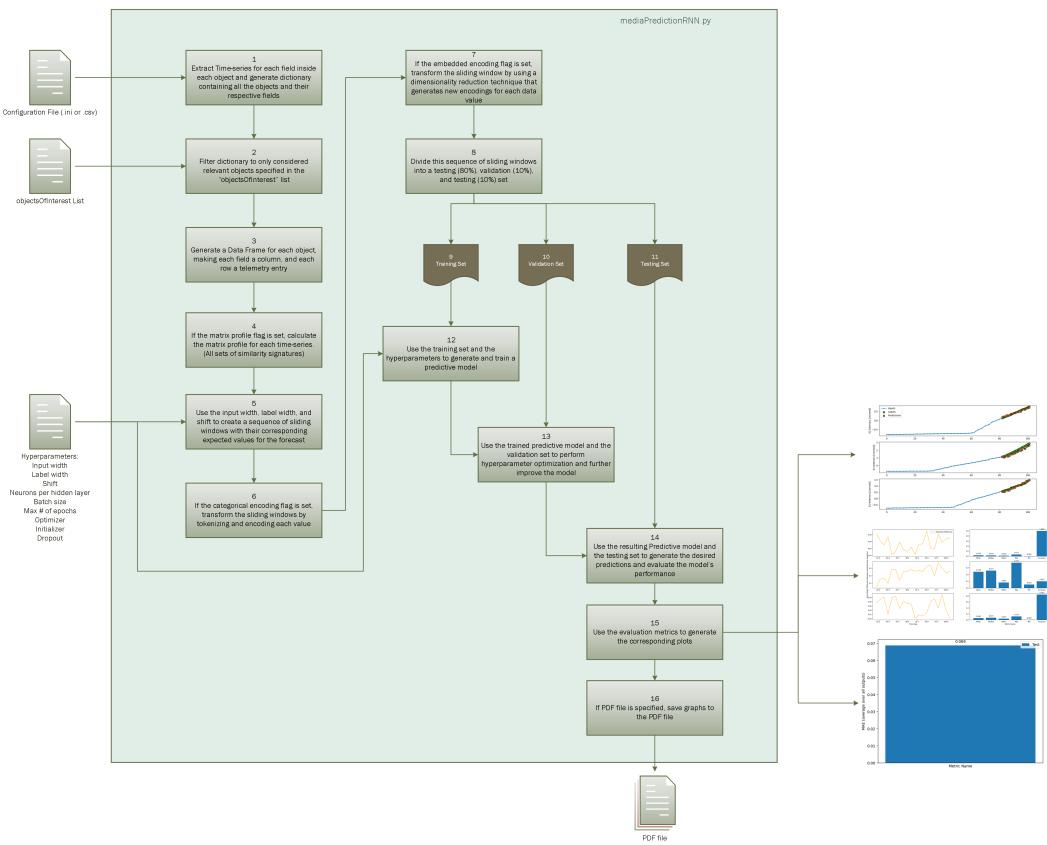
Phase 6

Phase 6: Perform Stat prediction



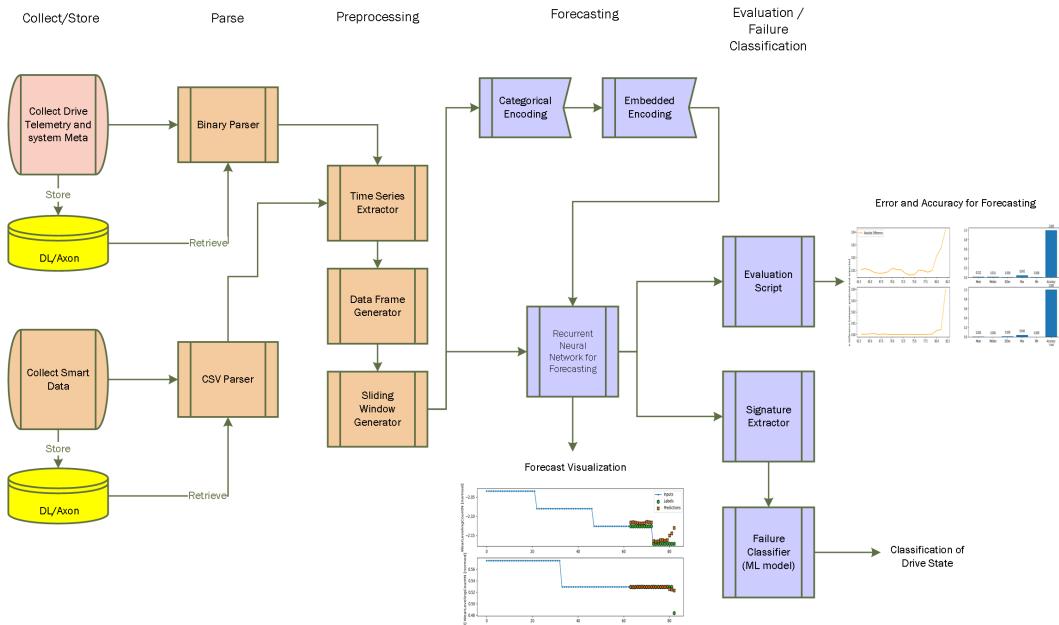
Phase 7

Phase 7: Run Predictive model



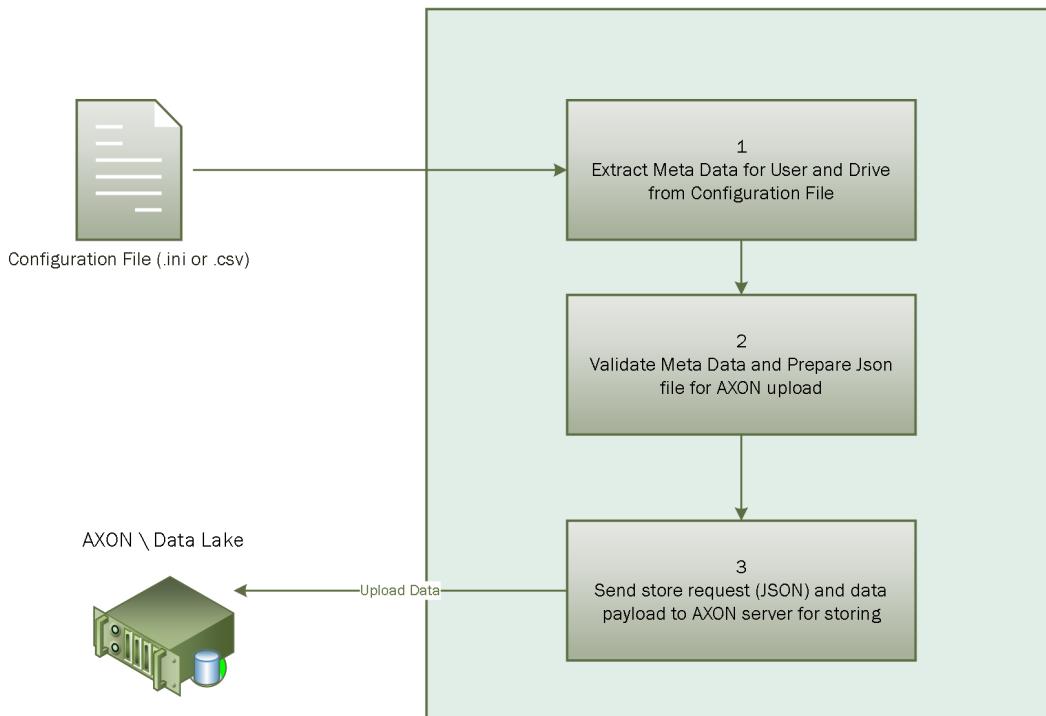
Phase 7 Appendix

RAAD Data Flow (Expert Context Report)



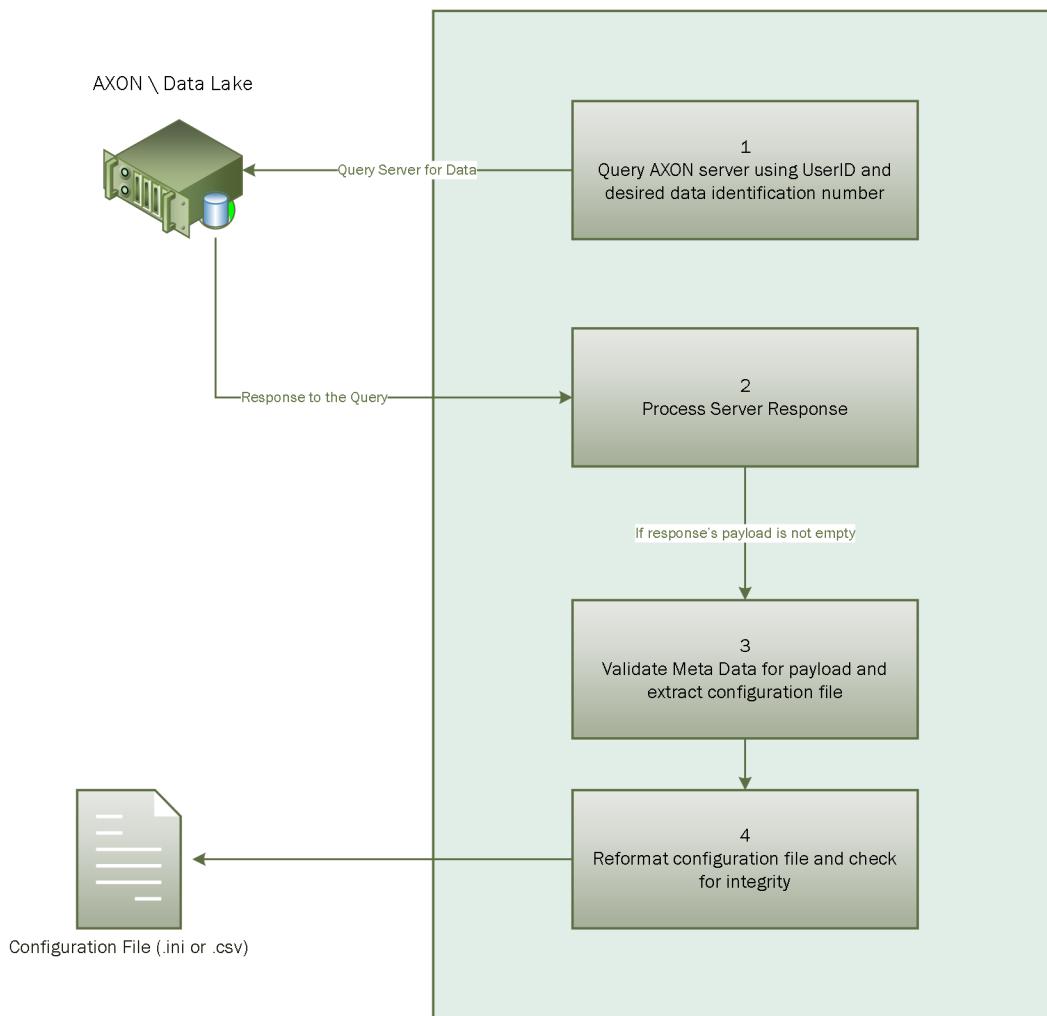
Phase 8

Phase 8: Upload content to AXON

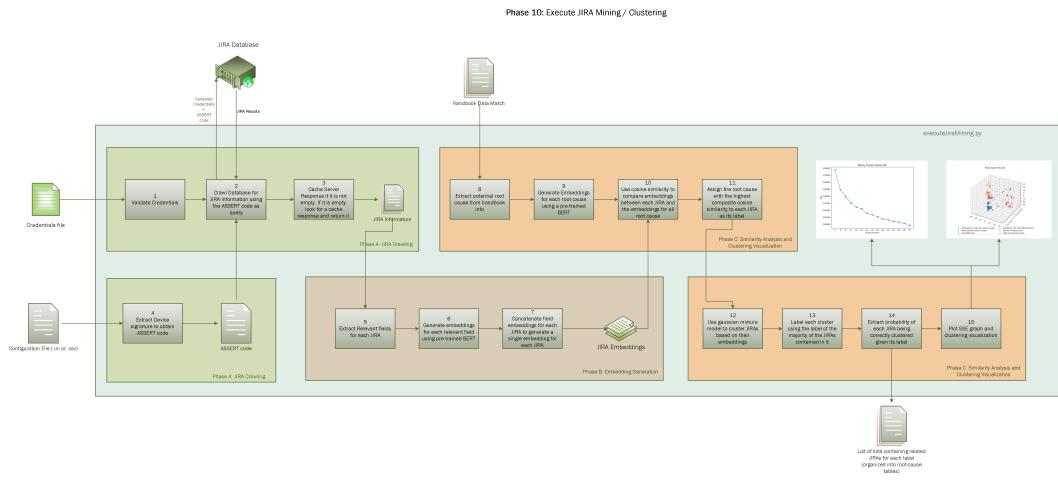


Phase 9

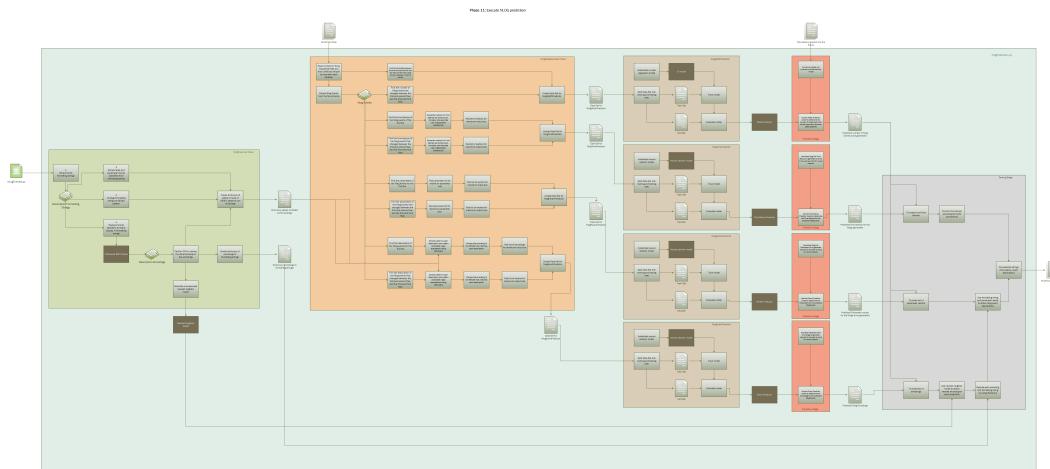
Phase 9: Download content from AXON



Phase 10

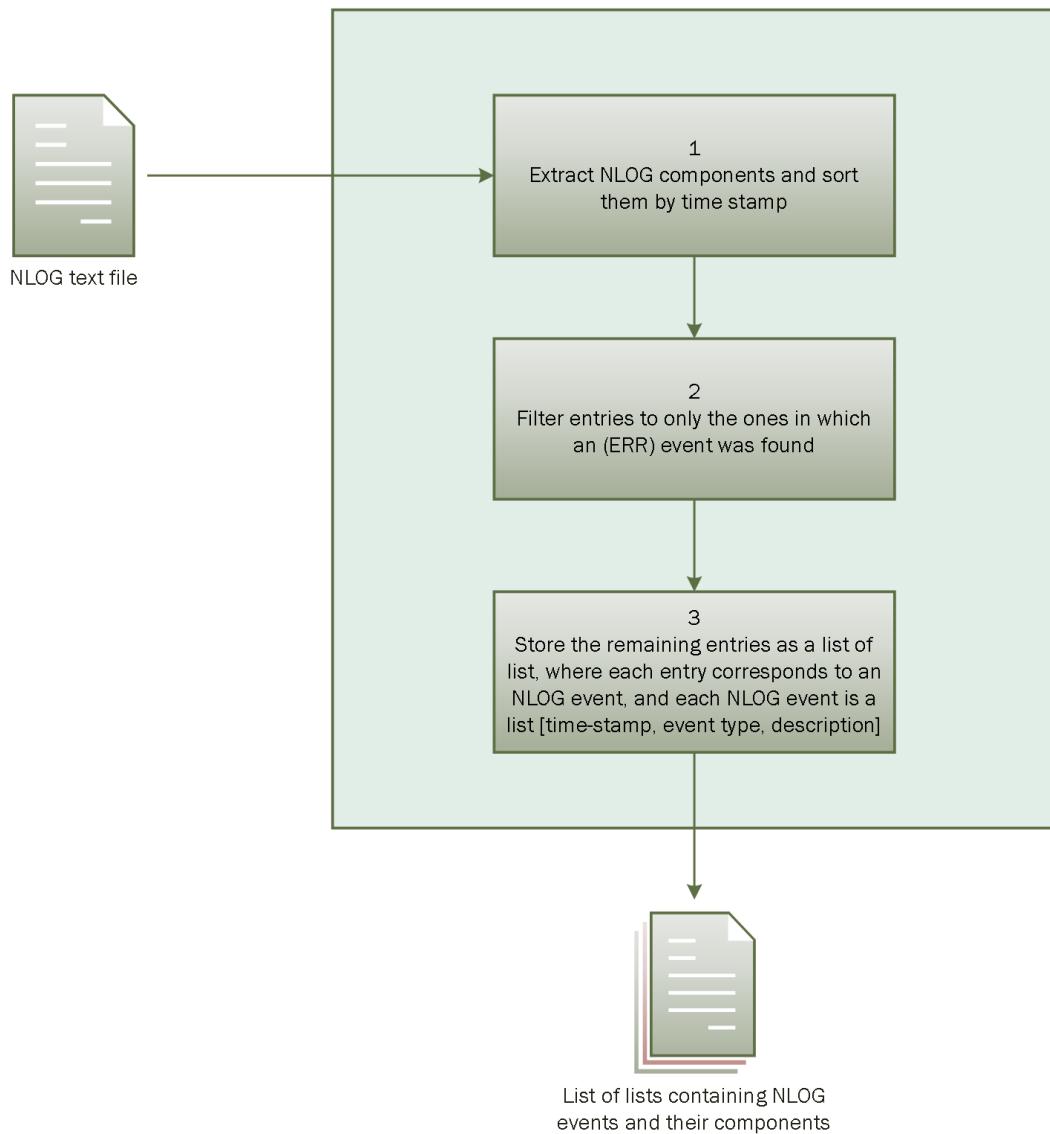


Phase 11



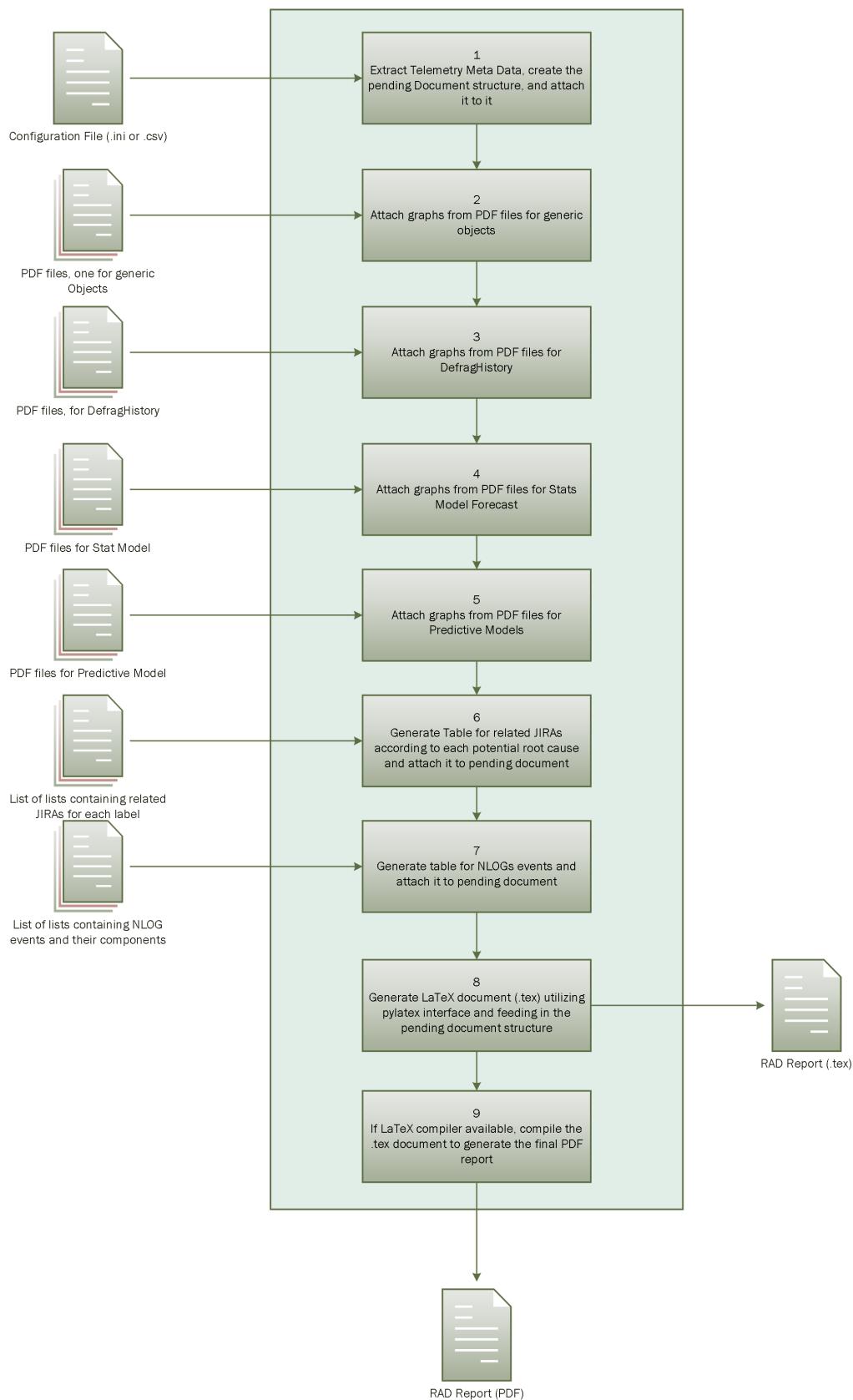
Phase 12

Phase 12: Extract NLOG table

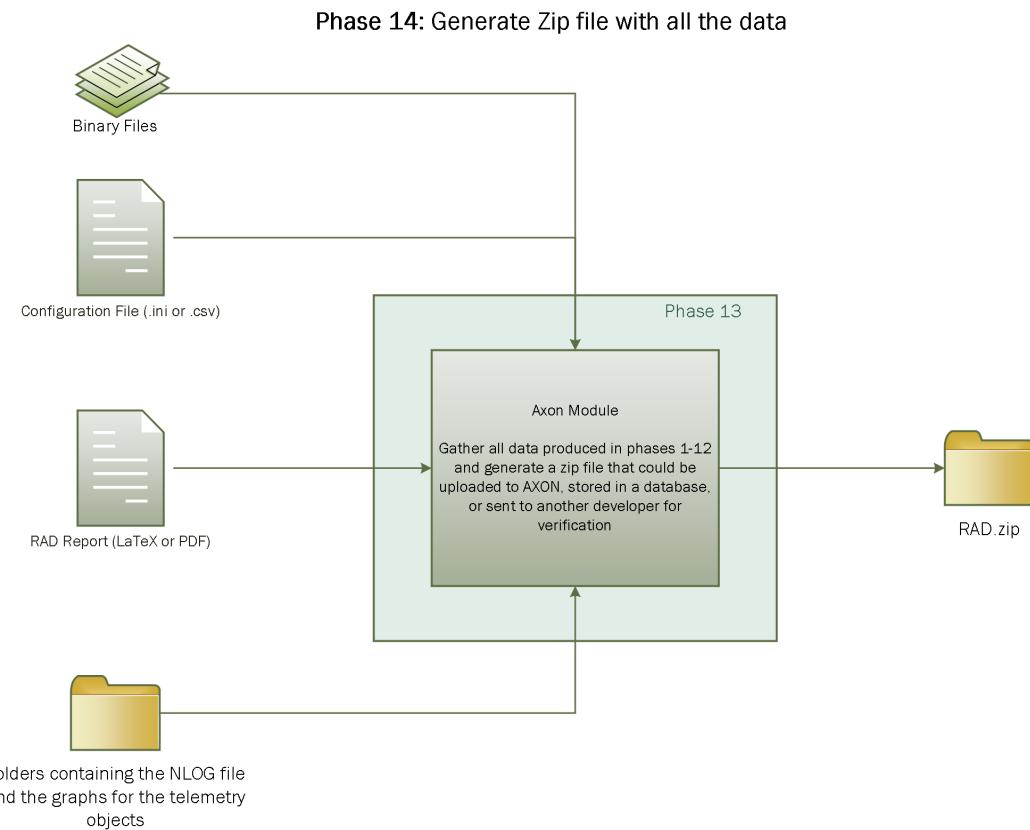


Phase 13

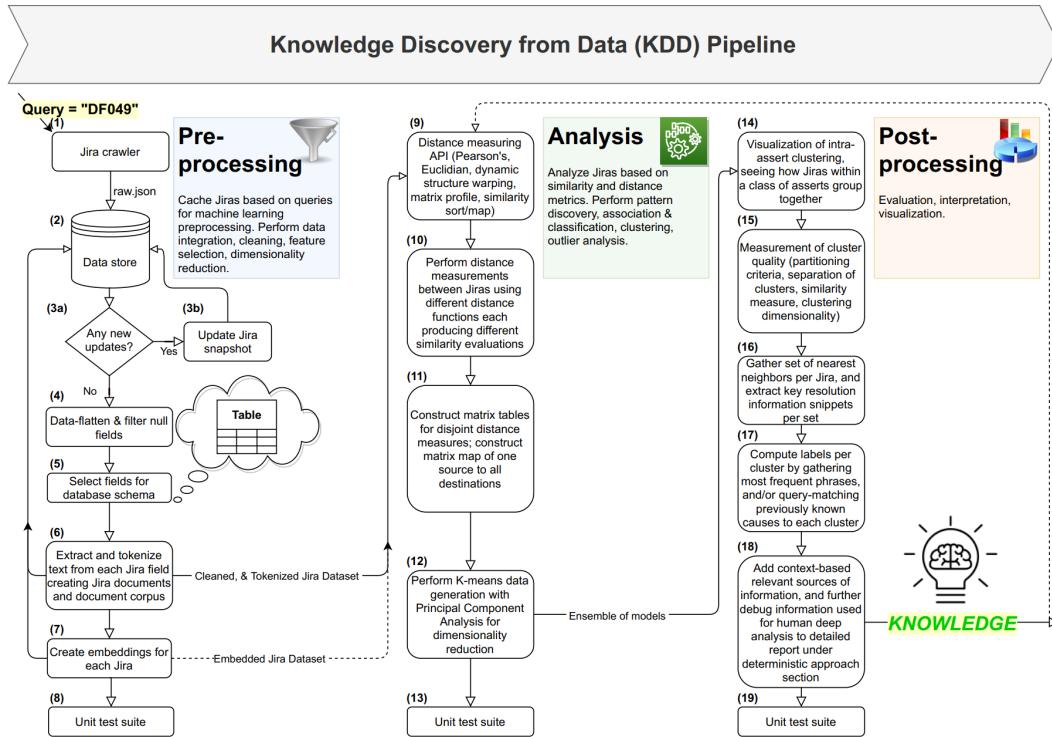
Phase 13: Gather Meta Data and Generate report



Phase 14



Wiki Root Cause



RAAD Server with Ubuntu 18+ x64_86 (Linux)

The persistent memory server is online and ready for larger Artificial Intelligence/Machine Learning workloads. Users may need to supplement power (~350 Watts) for an Nvidia RTX A6000. The server power should be physically operated or through Intelligent Platform Management Interface (IPMI). We can check the control server by ssh with username@server. Lastly, the BIOS on the server can share the network through one port. Ideally, a user want to we have redundant ethernet cables in the case of one switch failing.

Special Thanks to Steve Scargall (Intel)

Server Platform Recommendation

2 Socket Wolfpass Server (Purley Platform) using the S2600WF motherboard CPU

- Support Requires: Xeon Cascade Lake CPU 2nd generation Scalable
- Recommended: Cascade Lake Platinum 8260L or 8280L
- Note: CPUs without letters, some have the 'M' notation, and some are 'L'. If there's no letter, that CPU supports up to 1TiB per socket (DDR+DCPMM) (Small), the 'M' (Medium) SKUs support up to 2TiB per socket, and the 'L' (Large) SKUs support up to 4.5TiB. The T, V, and Y CPU SKUs have additional features that are unrelated to DCPMM.

DIMM Recommended Configurations

- Minimum Required
 - 8 Optane per 1 DRAM
- Configurations
 - 12 x DDR (16GB, 32GB, or 64GB)
 - 12 x DCPMM (128GB, 256GB, or 512GB)

- Acutal
 - 1.5 TB Optane and 192 DRAM

Operating Systems:

- Linux Recommendation: Kernel version 4.19 or later
- Ubuntu 18.1x+ LTS

File Systems

- Ext4
- ZFS

Git Repos

- <https://github.com/intel/ipmctl>

Training

- Explanation Video
 - <https://software.intel.com/content/www/us/en/develop/videos/provisioning-intel-optane-dc-persistent-memory-modules-in-linux.html>
- Supermicro Documentation
 - https://www.supermicro.com/support/resources/memory/DCP_MM_1stGen_memory_config_purley.pdf

Memory Installation and Population

DDR4 and Intel Optane Persistent Memory must be installed in accordance with the memory population guidelines. Refer to the Intel Server Board S2600WF Product Family Technical Product Specification document for detailed information.

- https://www.intel.com/content/dam/support/us/en/documents/server-products/server-boards/S2600WF_TPS.pdf

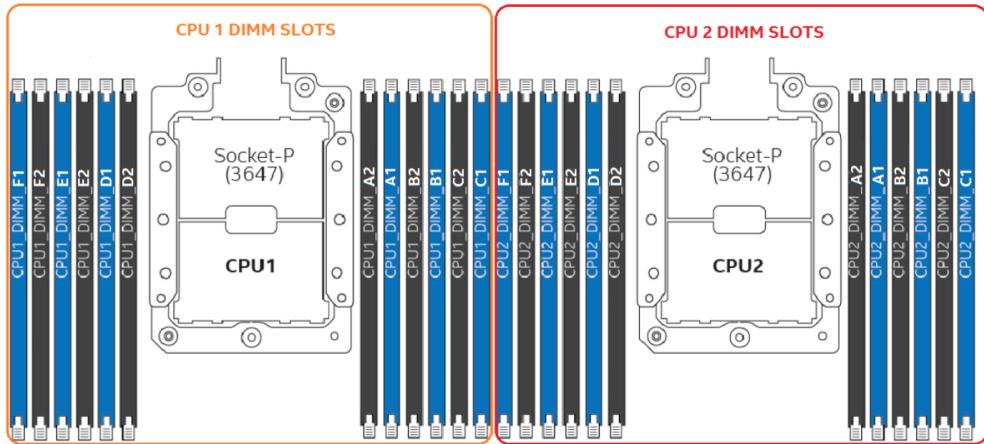


Figure 29. Intel® Server Board S2600WF memory slot layout

Physical memory slot identifiers

Table 16. Traditional DRAM DIMM-only population configurations

# of DIMMs	iMC1						iMC0					
	CH F		CH E		CH D		CH C		CH B		CH A	
	Slot 2	Slot 1										
1	-	-	-	-	-	-	-	-	-	-	-	DRAM
2	-	-	-	-	-	-	-	-	-	DRAM	-	DRAM
3	-	-	-	-	-	-	-	DRAM	-	DRAM	-	DRAM
4	-	-	-	DRAM	-	DRAM	-	-	-	DRAM	-	DRAM
5 ¹	-	-	-	DRAM	-	DRAM	-	DRAM	-	DRAM	DRAM	DRAM
6	-	DRAM										
7 ¹	-	DRAM	DRAM	DRAM								
8	-	-	DRAM	DRAM	DRAM	DRAM	-	-	DRAM	DRAM	DRAM	DRAM
9 ¹	-	DRAM	-	DRAM	-	DRAM						
10 ¹	-	DRAM	DRAM	DRAM	DRAM	DRAM	-	DRAM	DRAM	DRAM	DRAM	DRAM
11 ¹	-	DRAM										
12	DRAM											

¹ Configuration not recommended. This is an unbalanced configuration which will yield less than optimal performance.

Memory slot population when using only DDR4 memory. The table shows a single CPU socket population matrix. It is highly recommended to install memory using the same population across all CPU sockets in the system.

Table 17. Traditional DRAM DIMM + Intel® Optane™ DC persistent memory module population configurations

Modes	Symmetric Population per Processor Socket													
	iMC1				iMC0									
	Channel F		Channel E		Channel D		Channel C		Channel B		Channel A			
Slot 2	Slot 1	Slot 2	Slot 1	Slot 2	Slot 1	Slot 2	Slot 1	Slot 2	Slot 1	Slot 2	Slot 1	Slot 2	Slot 1	
AD	PMM	DRAM1	PMM	DRAM1	PMM	DRAM1	PMM	DRAM1	PMM	DRAM1	PMM	DRAM1	2-2-2	
MM	PMM	DRAM1	PMM	DRAM1	PMM	DRAM1	PMM	DRAM1	PMM	DRAM1	PMM	DRAM1	2-2-2	
AD + MM	PMM	DRAM3	PMM	DRAM3	PMM	DRAM3	PMM	DRAM3	PMM	DRAM3	PMM	DRAM3	2-2-2	
AD	-	DRAM1	-	DRAM1	PMM	DRAM1	-	DRAM1	-	DRAM1	PMM	DRAM1	2-1-1	
MM	-	DRAM2	-	DRAM2	PMM	DRAM2	-	DRAM2	-	DRAM2	PMM	DRAM2	2-1-1	
AD + MM	-	DRAM3	-	DRAM3	PMM	DRAM3	-	DRAM3	-	DRAM3	PMM	DRAM3	2-1-1	
AD	-	DRAM1	PMM	DRAM1	PMM	DRAM1	-	DRAM1	PMM	DRAM1	PMM	DRAM1	2-2-1	
MM	-	DRAM1	PMM	DRAM1	PMM	DRAM1	-	DRAM1	PMM	DRAM1	PMM	DRAM1	2-2-1	
AD + MM	-	DRAM3	PMM	DRAM3	PMM	DRAM3	-	DRAM3	PMM	DRAM3	PMM	DRAM3	2-2-1	
AD	-	PMM	-	DRAM1	-	DRAM1	-	PMM	-	DRAM1	-	DRAM1	1-1-1	
MM	-	PMM	-	DRAM1	-	DRAM1	-	PMM	-	DRAM1	-	DRAM1	1-1-1	
AD	-	PMM	DRAM1	DRAM1	DRAM1	DRAM1	-	PMM	DRAM1	DRAM1	DRAM1	DRAM1	2-2-1	

Note: AD = App Direct Mode and MM=Memory Mode

Shows the memory slot population matrix when using DDR4 and Intel Optane Persistent Memory modules. The table shows a single CPU socket population matrix. It is highly recommended to install memory using the same population across all CPU sockets in the system.

A fully populated system (2-2-2) should install DDR into Slot 1 (Blue) and Optane PMem in Slot 2 (Black) of every memory channel. A correctly installed configuration should have a population with the following (DDR/PMem capacity does not matter):

```
$ ipmctl show -topology
DimmID | MemoryType | Capacity | PhysicalID| DeviceLocator
=====
=====
0x0001 | Logical Non-Volatile Device | 252.438 GiB |
0x0026 | CPU1_DIMM_A2
0x0011 | Logical Non-Volatile Device | 252.438 GiB |
0x0028 | CPU1_DIMM_B2
0x0021 | Logical Non-Volatile Device | 252.438 GiB |
0x002a | CPU1_DIMM_C2
0x0101 | Logical Non-Volatile Device | 252.438 GiB |
0x002c | CPU1_DIMM_D2
0x0111 | Logical Non-Volatile Device | 252.438 GiB |
0x002e | CPU1_DIMM_E2
0x0121 | Logical Non-Volatile Device | 252.438 GiB |
```

```

0x0030 | CPU1_DIMM_F2
0x1001 | Logical Non-Volatile Device | 252.438 GiB |
0x0032 | CPU2_DIMM_A2
0x1011 | Logical Non-Volatile Device | 252.438 GiB |
0x0034 | CPU2_DIMM_B2
0x1021 | Logical Non-Volatile Device | 252.438 GiB |
0x0036 | CPU2_DIMM_C2
0x1101 | Logical Non-Volatile Device | 252.438 GiB |
0x0038 | CPU2_DIMM_D2
0x1111 | Logical Non-Volatile Device | 252.438 GiB |
0x003a | CPU2_DIMM_E2
0x1121 | Logical Non-Volatile Device | 252.438 GiB |
0x003c | CPU2_DIMM_F2
N/A | DDR4 | 32.000 GiB | 0x0025 | CPU1_DIMM_A1
N/A | DDR4 | 32.000 GiB | 0x0027 | CPU1_DIMM_B1
N/A | DDR4 | 32.000 GiB | 0x0029 | CPU1_DIMM_C1
N/A | DDR4 | 32.000 GiB | 0x002b | CPU1_DIMM_D1
N/A | DDR4 | 32.000 GiB | 0x002d | CPU1_DIMM_E1
N/A | DDR4 | 32.000 GiB | 0x002f | CPU1_DIMM_F1
N/A | DDR4 | 32.000 GiB | 0x0031 | CPU2_DIMM_A1
N/A | DDR4 | 32.000 GiB | 0x0033 | CPU2_DIMM_B1
N/A | DDR4 | 32.000 GiB | 0x0035 | CPU2_DIMM_C1
N/A | DDR4 | 32.000 GiB | 0x0037 | CPU2_DIMM_D1
N/A | DDR4 | 32.000 GiB | 0x0039 | CPU2_DIMM_E1
N/A | DDR4 | 32.000 GiB | 0x003b | CPU2_DIMM_F1

```

Updating the BIOS and Intel Optane Persistent Memory Firmware

1. This procedure is for Intel whitebox servers only (aka Wolfpass). This will not work on OEM systems.
2. Use a FAT32 formatted USB stick/thumb drive to store BIOS image.
3. If the USB drive is local to your laptop/desktop and will be inserted into the remote server after creating the USB drive, perform steps 2a-2d. If the server is remote with a USB drive inserted, perform steps 2e-2k.
4. If the system is currently running a pre-production or debug bios, you must put the system in the BIOS Recovery mode otherwise you'll end up with errors similar to those shown in Figure 1. If it running a

production BIOS and we are simply updating it, skip the BIOS recovery mode step.

5. A production BIOS version is 02.01.0001 or later, eg:
SE5C620.86B.02.01.0010.010620200716

DO NOT remove the USB flash drive

=====

Flashing ME FW...

System BIOS and ME Update Utility Version 14.1 Build 14
Copyright (c) 2019 Intel Corporation

Processing Capsule File - R02010010_Production_ACM_TXT_ME.signed.cap

ME Update In Progress:

Error: The capsule file in the update package is invalid

Error: BIOS interface failed - this can occur while reading / writing to BIOS

Echo is off.

comp: File not found - 'reset.log'

SUT will auto reboot later for BIOS/BMC/FRUSDR taking effect.

-

How To enter BIOS Recovery Mode

To put the system into BIOS Recovery mode do the following:

- Disconnect AC, move the white jumper J5A3 (BIOS_RCVR), and black jumper J5A4 (ME_FCR_UPDT), behind the video connector, from 1-2 to 2-3.
 - Power the system up, it will boot to recovery mode
 - Press F6 to enter the Boot Manager at boot time
 - Select 'Launch EFI Shell' from the list
 - At UEFI Shell, run the update scripts from the BIOS package (See 'BIOS ad Firmware Update Procedure' section)
 - When done, disconnect AC, move both jumpers J5A3 and J5A4 back to 1-2

- Boot the system normally

BIOS and Firmware Update Procedure

1. Download the latest zip file for Production servers from
<https://downloadcenter.intel.com/download/29341/Intel-Server-Board-S2600WF-Family-BIOS-and-Firmware-Update-Package-for-UEFI>
2. If the USB is to be created locally on a laptop/desktop then inserted into the target system:
 - Format the USB Drive using FAT32
 - Unzip the BIOS package
 - Copy the contents to the root of the USB drive
 - Remove the USB Drive and insert into the target system
 - Else, if the USB drive is already inserted into the target system, perform the following:
 - Copy the zip file to the target system (we'll assume to /downloads for this procedure)
 - Unzip the file, eg:
 - \$ unzip
 S2600WF_EFI_BIOSR02010010_ME04.01.04.339
 _BMC2.37.1f190479_FRUSDR_1.98_DCPMM01.
 02.00.5417.zip
 - Identify the USB drive (Assumes /dev/sde as an example)

```
$ lsblk -l
NAME                MAJ:MIN RM  SIZE
RO  TYPE MOUNTPOINT
```

```
sde          8:64      1  29.9G
0 disk
```

- Remove any existing partitions

```
$ wipefs -a /dev/sde
```

- Format the USB Drive using Fat32 (vfat):

```
$ mkfs.vfat -F32 /dev/sde
```

- Mount the USB Drive

```
$ mkdir /mnt/usbdrive
$ mount -t vfat /dev/sde /mnt/usbdrive
```

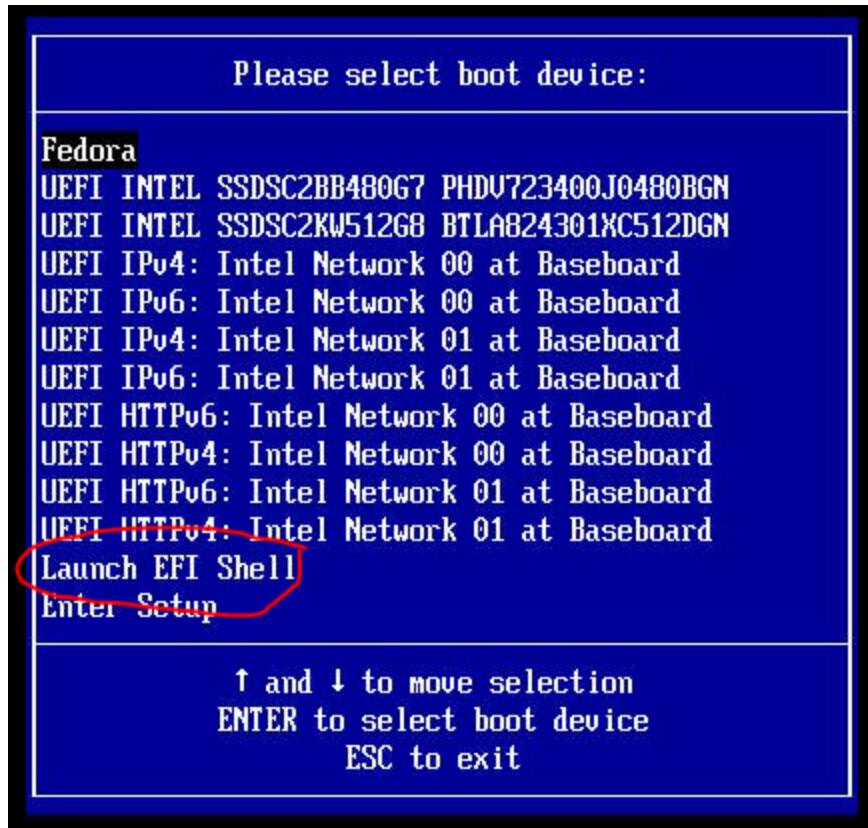
- Copy the contents of the ZIP file to the USB Drive

```
$ cd
/downloads/S2600WF_EFI_BIOSR02010010_ME
04.01.04.339_BMC2.37.1f190479_FRUSDR_1.
98_DCPMM01.02.00.5417/
$ cp -rp * /mnt/usbdrive
$ sync
$ umount /mnt/usbdrive
```

3. Connect to the server through the BMC (or serial)

4. Insert the USB stick into the target server & reboot

5. When you see the BIOS , press F6 to enter the Boot Manager and select “Launch EFI Shell” from the list:



6. If the BIOS/Firmware image contents are located in the root of the USB, the BIOS should execute the 'startup.nsh' script located in the root directory.
 - If the BIOS/Firmware image contents are located in a directory on the USB Drive:
 - Select the USB Drive. This will usually be 'FS0:' or 'FS1'
 - From the prompt, type 'FS0:' (without quotes. The colon is important)
 - Change directory to the folder with the BIOS/Firmware image, eg

```
> cd WFP_SUP_WW37
```
 - List the contents and verify the 'startup.nsh' script is available

```
> ls
```

- Execute the 'startup.nsh' script

```
> startup.nsh
```

7. After a short initialization phase, the script will stop and prompt for input. It shows the 6 steps and requests for you to press any key to continue
8. Steps 1-5 will be performed.
 - Note: If you connect through the BMC web interface or SSH, the screen may go black and you will lose connection. This is because the BMC update restarts all connections. Just close your session/window and re-connect.
9. The script will then ask you to update the SDR and FRU.
 - Select option 3 from the list.
 - Note: You may see 'FRU Board P/N is invalid!' which is okay
10. When the update completes, the system will automatically reboot.
11. Let the system boot to allow the firmware updates to complete. Note: The system may reboot several times within the BIOS before it actually tries to boot into the OS. This is okay.
12. [Optional] If DCPMM is being installed or reconfigured, power off the host and physically install the DDR & DCPMM Memory. Otherwise, skip to Step 18
 - For my 2S 2U Wolfpass, DCPMM is installed in the Black DIMM Slots - CPU[12]_DIMM_[A-F]2 and the DDR in the '1' slots (Blue DIMM Slots). There should be a DIMM location diagram on the top of the server to verify this before installing. Here's the topology output from my lab system so you can map the DCPMM and DDR slot locations:

```
$ ipmctl show -topology
DimmID | MemoryType | Capacity
| PhysicalID | DeviceLocator
0x0001 | Logical Non-Volatile Device | 252.4
GiB | 0x0028 | CPU1_DIMM_A2
0x0011 | Logical Non-Volatile Device | 252.4
GiB | 0x002c | CPU1_DIMM_B2
0x0021 | Logical Non-Volatile Device | 252.4
GiB | 0x0030 | CPU1_DIMM_C2
0x0101 | Logical Non-Volatile Device | 252.4
GiB | 0x0036 | CPU1_DIMM_D2
0x0111 | Logical Non-Volatile Device | 252.4
GiB | 0x003a | CPU1_DIMM_E2
0x0121 | Logical Non-Volatile Device | 252.4
GiB | 0x003e | CPU1_DIMM_F2
0x1001 | Logical Non-Volatile Device | 252.4
GiB | 0x0044 | CPU2_DIMM_A2
0x1011 | Logical Non-Volatile Device | 252.4
GiB | 0x0048 | CPU2_DIMM_B2
0x1021 | Logical Non-Volatile Device | 252.4
GiB | 0x004c | CPU2_DIMM_C2
0x1101 | Logical Non-Volatile Device | 252.4
GiB | 0x0052 | CPU2_DIMM_D2
0x1111 | Logical Non-Volatile Device | 252.4
GiB | 0x0056 | CPU2_DIMM_E2
0x1121 | Logical Non-Volatile Device | 252.4
GiB | 0x005a | CPU2_DIMM_F2
N/A | DDR4 | 32.0 GiB
| 0x0026 | CPU1_DIMM_A1
N/A | DDR4 | 32.0 GiB
| 0x002a | CPU1_DIMM_B1
N/A | DDR4 | 32.0 GiB
| 0x002e | CPU1_DIMM_C1
N/A | DDR4 | 32.0 GiB
| 0x0034 | CPU1_DIMM_D1
N/A | DDR4 | 32.0 GiB
| 0x0038 | CPU1_DIMM_E1
N/A | DDR4 | 32.0 GiB
| 0x003c | CPU1_DIMM_F1
N/A | DDR4 | 32.0 GiB
| 0x0042 | CPU2_DIMM_A1
N/A | DDR4 | 32.0 GiB
| 0x0046 | CPU2_DIMM_B1
```

N/A	DDR4	32.0 GiB
0x004a	CPU2_DIMM_C1	
N/A	DDR4	32.0 GiB
0x0050	CPU2_DIMM_D1	
N/A	DDR4	32.0 GiB
0x0054	CPU2_DIMM_E1	
N/A	DDR4	32.0 GiB
0x0058	CPU2_DIMM_F1	

13. Power on the host. Note: The system can take a VERY long time to get past the CPU and Memory Initialization. Perhaps up to 10mins on the 6TB (512GB) system.
14. Assuming no issues, let the system boot into the OS to verify no issues are encountered with the new hardware
15. Assuming no issues, reboot the host and go back in to the UEFI Shell (F6 during the BIOS screen) [Repeat Step 7]
16. From the UEFI Shell prompt, change to the USB Drive using 'fs0:' (Note: the colon is important!) [Repeat Step 8]

Shell> fs0:

(This could also be 'fs1:' depending on the number of drives)

17. You should see the 'WFP_SUP_WW37' directory [Repeat Step 9]

```
FS0:> ls
WFP_SUP_WW37
> cd WFP_SUP_WW37
```

18. Run the 'startup.nsh' script located in the directory [Repeat Step 10]

```
FS0:\WFP_SUP_WW37\> startup.nsh
```

19. This will detect that Steps 1-5 have been completed and will proceed with Step 6 (Update DCPMM Firmware)

```
Found DCPMM DIMM installed
Updating DCPMM firmware

=====
Continuing DCPMM Firmware update.....  
DO NOT remove the USB flash drive  
Starting update on 12 dimm(s) ...
Load FW on DIMM 0x0001: Success, a power cycle is required to activate the FW.
Load FW on DIMM 0x0011: Success, a power cycle is required to activate the FW.
Load FW on DIMM 0x0021: Success, a power cycle is required to activate the FW.
Load FW on DIMM 0x0101: Success, a power cycle is required to activate the FW.
Load FW on DIMM 0x0111: Success, a power cycle is required to activate the FW.
Load FW on DIMM 0x0121: Success, a power cycle is required to activate the FW.
Load FW on DIMM 0x1001: Success, a power cycle is required to activate the FW.
Load FW on DIMM 0x1011: Success, a power cycle is required to activate the FW.
Load FW on DIMM 0x1021: Success, a power cycle is required to activate the FW.
Load FW on DIMM 0x1101: Success, a power cycle is required to activate the FW.
Load FW on DIMM 0x1111: Success, a power cycle is required to activate the FW.
Load FW on DIMM 0x1121: Success, a power cycle is required to activate the FW.
```

- Note: If the script bails out with an error that the logs do not match, you can fix this using 'echo 1 > reset.log' then re-run 'startup.nsh' as show below:

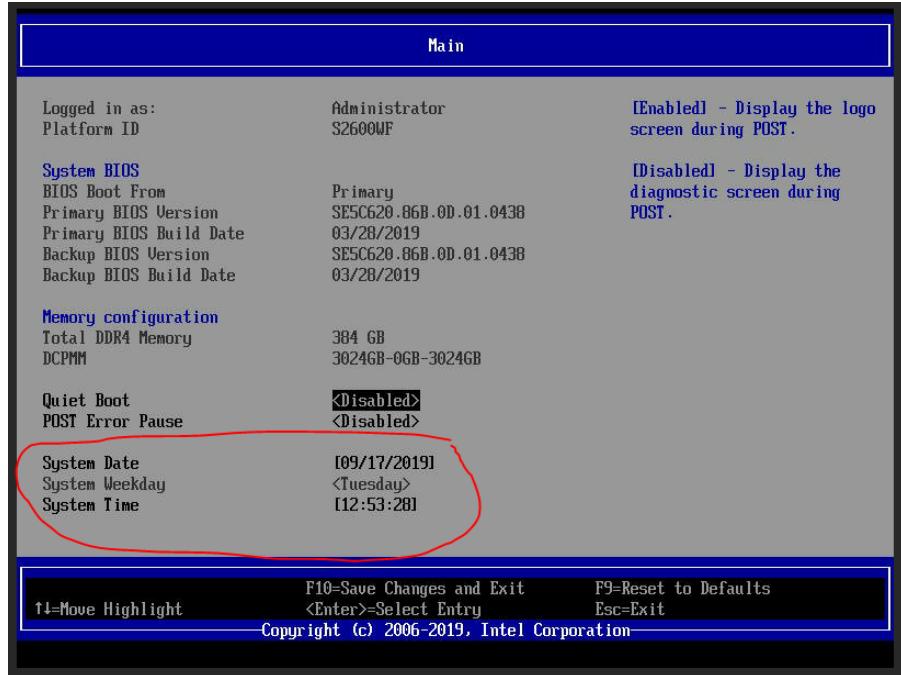
```
Enter 'q' to quit, any other key to continue:  
  
Compare firstresettemplate.log to reset.log.  
Difference # 1: File size mismatch.  
SUT will auto reboot later for BIOS/BMC/FRUSDR taking effect.  
Unable to redirect file.  
Script Error Status: Invalid Parameter (line number 126)  
FS0:\WFP_SUP_WW37\> ls *.log  
Directory of: FS0:\WFP_SUP_WW37\*.log  
09/17/2019 12:09 8 firstresettemplate.log  
09/17/2019 12:09 84 noDCPMM.log  
09/17/2019 12:31 1,802 CacheRiver.log  
09/17/2019 12:32 0 reset.log  
 4 File(s) 1,894 bytes  
 0 Dir(s)  
FS0:\WFP_SUP_WW37\> cat firstresettemplate.log  
1  
  
FS0:\WFP_SUP_WW37\> cat reset.log  
  
FS0:\WFP_SUP_WW37\> echo 1 > reset.log  
FS0:\WFP_SUP_WW37\> cat reset.log  
1  
  
FS0:\WFP_SUP_WW37\> startup.nsh
```

20. Once successful, you should be returned to the UEFI Shell prompt

```
Delete successful.  
Deleting 'F30:\WFP_SUP_WW37\BIOS_pass.txt'  
Delete successful.  
Deleting 'F30:\WFP_SUP_WW37\ME_pass.txt'  
Delete successful.  
Deleting 'F30:\WFP_SUP_WW37\FB_pass.txt'  
Delete successful.  
Deleting 'F30:\WFP_SUP_WW37\Frusdr_pass.txt'  
Delete successful.  
Deleting 'F30:\WFP_SUP_WW37\reset.log'  
Delete successful.  
Deleting 'F30:\WFP_SUP_WW37\ueBCPMM.txt'  
Delete successful.  
Deleting 'F30:\WFP_SUP_WW37\comp.txt'  
Delete successful.  
=====  
All system updates have now completed!  
  
Please note that this update process will erase the logs generated from the BMC/  
BIOS/ME/FD flash.  
You can now remove the USB Key and reboot the system.  
During POST access the BIOS Setup utility <F2> to confirm all updates have insta  
lled properly.  
F30:\WFP_SUP_WW37\> _
```

21. Type 'reset -c' to reset the host and clear the CMOS

22. Press F2 to enter the BIOS setup. Clearing the CMOS often resets the date/time so we need to fix this before allowing the OS to boot. The date/time settings are in the 'Main' menu of the BIOS (the default selected option).



23. Press F10 to save the changes and exit the BIOS

24. Allow the system to boot in to the OS

25. End of action plan.

Compatible Operating Systems for Intel® Optane™ Persistent Memory

Intel® Optane™ persistent memory operating system (OS) mode support information.

- The operating systems listed below have been verified by Intel and do not reflect the OS vendor support.
- Please contact the respective OS vendor(s) for the exact release version providing the proper support.
- For OS that are not listed, use open source code to generate necessary files.

Operating System Support

OS Version	Memory	Mode	Mode
	App	Direct	Dual Mode
RHEL* 7.5	Yes		
Ubuntu* 16.04 LTS	Yes		
Windows* Server 2016	Yes		
Oracle* Linux* 7.6 with UEK R5 Update 2	Yes	Yes	
VMware* vSphere 6.7 EP10	Yes	Yes	
CentOS* 7.6 or later	Yes	Yes	Yes
RHEL 7.6 or later	Yes	Yes	Yes
SLES* 12 SP4 or later	Yes	Yes	Yes
SLES 15 or later	Yes	Yes	Yes
Ubuntu 18.04 LTS	Yes	Yes	Yes
Ubuntu 18.1+ LTS	Yes	Yes	Yes

OS Version	Memory	Mode	Mode
	Direct	App	Dual Mode
VMWare* ESXi 6.7 U1 or later	Yes	Yes	Yes
Windows 10 Pro for Workstation Version 1809 or later	Yes	Yes	Yes
Windows Server 2019 or later	Yes	Yes	Yes

Platform Details

Intel® Xeon® Platinum 8260L CPU & Optane™ DC 512GB Persistent Memory Supermicro 2U/4S 64GB {AEP}

- Processors 4 x Intel® Xeon® Platinum 8260L Processor (35.75M Cache, 2.40 GHz)
- Memory 24 x 64GB 2933MHz PC4-23400 ECC Registered 1.2 Volts DDR4 DIMM
- Memory 24 x 512GB Apache Pass 3D XPoint Persistent Memory DIMM [NMA1XXD512GPSU4]
- Storage 1 x 240GB Intel® SSD DC S4510 Series (Youngsville Refresh) SATA 6Gb/s 2.5"
- Video Adapter 1 x Aspeed AST2500 On-Board
- Remote Management 1 x SuperMicro Intelligent Management (IPMI 2.0)
- Operating System 1 x Ubuntu 18.1x+ LTS
- Power Supplies 2 x 1600 Watts Hot-Swappable 110/220v
- Rails 1 x Supermicro Ball bearing Rails MCP-290-00057-0N [In Box]

- Add-in Adapter 1 x Intel® Ethernet Converged Network Adapter X550-T2 10Gbe Dual-Port
- Onboard Adapter 1 x Intel® i350 1Gbe NIC on SuperMicro AOC-2UR66-i4G Riser Quad-Port

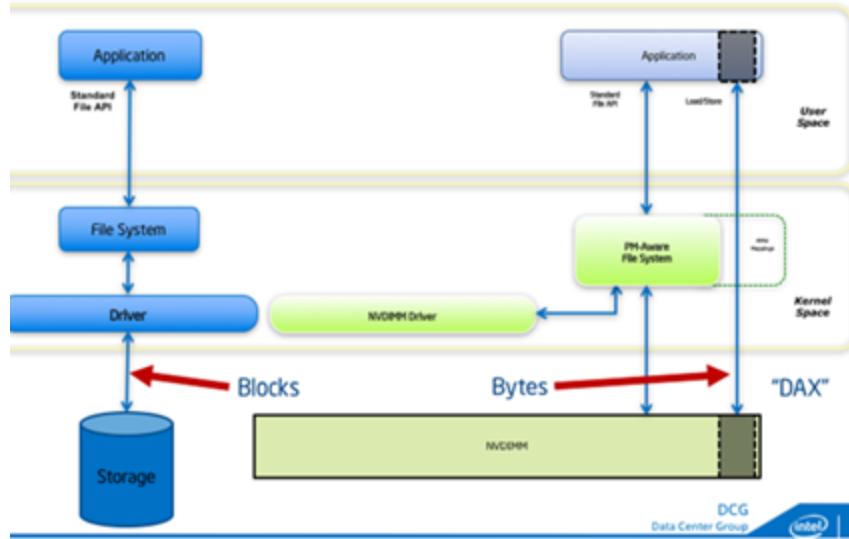
The model of system is:

- Supermicro SYS-2049U-TR4

SuperMicro Site

- <https://www.supermicro.com/en/products/system/2U/2049/SYS-2049U-TR4.cfm>
- <https://www.supermicro.com/products/system/2U/2029/SYS-2029U-TN24R4T.cfm>
- S/N: WO99274L02S004
- Model: 218U-16
- P/N: SYS-2049U-TR4
- PCODE: 2049U-TR4-IDD
- OS SATA drive: Intel D3-S4510 Series 240GB ISN: BTYF918004B4240AGN
- Note: units configuration will support 12 SATA3 (6Gbps) and 4 x U.2 Intel NVMe SSD NVMe drives in PCIe Slots only.

Persistent Memory Programming Model



Persistent Memory with direct access (DAX)

How to optimize Optane DIMM usage for DRAM Caching

Memkind: is a persistent memory mapped files aware to the file system. Details of using it <https://software.intel.com/en-us/articles/volatile-use-of-persistent-memory>

libvmemcache: is an embeddable and lightweight in-memory caching solution. It's designed to fully take advantage of large capacity memory, such as Persistent Memory with direct access (DAX), through memory mapping in an efficient and scalable way. Details of using it <https://docs.pmem.io/persistent-memory/getting-started-guide/what-is-pmdk>

Technical Details

1. Introduction to Persistent Memory Programming
 - o <https://software.intel.com/en-us/articles/introduction-to-programming-with-persistent-memory-from-intel>

2. DIMM usage: Overview information

- Brief
<https://www.intel.com/content/www/us/en/products/docs/memory-storage/optane-persistent-memory/optane-dc-persistent-memory-brief.html>
- Quick Start
[https://www.intel.com/content/dam/support/us/en/documents/memory-and-storage/Intel-MDT-setup-guide.pdf](https://www.intel.com/content/dam/support/us/en/documents/memory-and-storage/data-center-persistent-mem/Intel-Optane-DC-Persistent-Memory-Quick-Start-Guide.pdf)
- Configurations for optimized performance
<https://www.intel.com/content/dam/support/us/en/documents/memory-and-storage/intel-mdt-setup-guide.pdf>

3. Software

- Caching Software: Focuses on how to cache SSDs and potentially app direct mode servers
 - <https://open-cas.github.io/index.html>
- Storage Performance Development Kit – This is the intel supported development kit to support optimizing transactions.
 - <https://spdk.io/doc/>
- Distributed Object Storage Software (DAOS) Storage software stack to accelerate
 - <https://www.intel.com/content/dam/www/public/us/en/documents/solution-briefs/high-performance-storage-brief.pdf>
- Publication on academic studies of Optane DIMMs
 - Academic paper on Optane DIMMs Basic Information
 - <https://arxiv.org/pdf/1903.05714.pdf>
 - Optimizing Optane DIMMs Transactions
 - <http://www.cse.lehigh.edu/~spear/papers/zardoshti-ipdps-2020.pdf>
 - C++ LLVM transactions light weight support

- <http://www.cse.lehigh.edu/~spear/papers/zardoshti-taco-2019.pdf>

Ubuntu 18.1x+ LTS x86_64 Instructions

Operating system SATA SSD is to be partitioned in ext4 with the additional storage configured in ZFS pools. The Intel NVDIMM SW stack implements several aspects to enable NVDIMM:

- nvdimm driver[12]
- Direct Access (DAX) support with file system[14]
- Block Translation Table (BTT)[15]
- UEFI Human Interface Infrastructure (HII) Firmware
- ndctl[16]: Utility library for managing the libnvdimm (non-volatile memory device) sub-system in the Linux kernel
- ipmctl[17]: Utility for configuring and managing Intel Optane DC persistent memory modules

Install Commands

```
$ sudo add-apt-repository ppa:woodrow-shen/ppa
$ sudo apt-get update
$ sudo apt install -y ipmctl libipmctl-dev ledmon ndctl zfs-
initramfs zfsutils-linux zfs-initramfs libzfslinux-dev zfs-
auto-snapshot libvirt-daemon-driver-storage-zfs python3-pyzfs
pyzfs-doc golang-go-zfs-dev libgtk-3-dev golang git nfs-common
golang-github-gotk3-gotk3-dev btrfs-progs e2fsprogs f2fs-tools
dosfstools hfsutils hfsprogs jfsutils mdadm util-linux
cryptsetup dmsetup lvm2 util-linux nilfs-tools nilfs-tools
ntfs-3g ntfs-3g reiser4progs reiserfsprogs reiserfsprogs
udftools xfsprogs xfsdump gpart gedit samba rsync grsync rar
unrar p7zip-full p7zip-rar openconnect libncurses5 libtinfo5
libbz1 openvpn vpnc-scripts net-tools network-manager-openvpn
network-manager-l2tp-gnome postfix libsasl2-modules ca-
certificates mailutils ubuntu-mate-desktop mate-desktop-
environment-extras mate-tweak gnome-tweaks wine
$ wget
https://www.thawte.com/roots/thawte_Premium_Server_CA.pem
$ sudo mv thawte_Premium_Server_CA.pem
/etc/ssl/certs/Thawte_Premium_Server_CA.pem
```

```
$ sudo cat /etc/ssl/certs/Thawte_Premium_Server_CA.pem | sudo
tee -a /etc/postfix/cacert.pem
$ sudo apt-get upgrade -y
```

Performance Optimizations

```
# Do not use ext3 for the OS file system, use ext4!
$ sudo preload cpupower-gui indicator-cpufreq sysv-rc-conf
numad

# Modify the cpu power to be for performance
$ sudo sysctl vm.swappiness=1
$ swapoff -a

# Wait 20 minutes...
$ swapon -a

$ sudo gedit /etc/sysctl.conf
# Modify or add the value
    vm.swappiness=10
    vm.vfs_cache_pressure=50
    vm.dirty_background_ratio = 5
    vm.dirty_background_bytes = 0
    vm.dirty_ratio = 10
    vm.dirty_bytes = 0
    vm.dirty_writeback_centisecs = 500
    vm.dirty_expire_centisecs = 12000

$ sudo gedit /etc/fstab
# Add the following:
    # Move /tmp to RAM
    tmpfs /tmp tmpfs defaults,noexec,nosuid 0 0

$ sudo ufw logging off
$ sudo chmod +x /etc/rc.local

$ sudo gedit /etc/rc.local
# Add this above "exit 0", we want these to reflect the
performance of the media read-write Input-Output size.
# Please refer to [24] and domain expert for internal mechanics
of the storage device. The 'sda' is the disk location in Linux.
echo 0 > /sys/block/sda/queue/add_random
echo 0 > /sys/block/sda/queue/rotational
```

```
echo 2 > /sys/block/sda/queue/rq_affinity
echo 256 > /sys/block/sda/queue/nr_requests
echo 256 > /sys/block/sda/queue/read_ahead_kb
exit 0

$ sudo /etc/pam.d/common-session and /etc/pam.d/common-session-
noninteractive
# Add
session required pam_limits.so
# The total size should be a reflection of total system memory
with room for the operating system and background tasks.
# These are in KB for the file so ensure you convert properly.
# Note: the user is @perf refers to the performance user group
and user_ID is for a specific user. The '-' means soft=hard
limit.
# Ensure the system is calculated and configured for total
users by ensuring swap space is ready for all users operating
at peak usage.
# In our case we have, 24 units x 1 DIMM x 64 GB per DRAM =
1,536 GB DRAM and 6 units x 4 DIMMs x 512 GB per Optane DIMM =
12,288 GB.
# Total runtime memory is 13,824 GB ~ 13.8 TB.
# We will limit the average user to the floor power of two for
the DRAM size so:
# T = 1,536 GB ~= (1,536 * 1,048,576 KB) ~= 1610612736 KB
# log(1610612736)/log(2) ~= 30.5
# floor(30.5) ~= 30, (2^30)-1 = 1073741823, we subtract 1 for
unsigned Linux kernel representation.
# Note: the Linux kernel variable size is the limit, so for
developers check the value in the source code.
#       nofile is based on cat /proc/sys/fs/nr_open
user_ID soft nofile 32768
user_ID hard nofile 32768
user_ID soft fsiz 1073741823
user_ID hard fsiz 1073741823
user_ID soft data 1073741823
user_ID hard data 1073741823
user_ID soft stack 1073741823
user_ID hard stack 1073741823
user_ID soft rss 1073741823
user_ID hard rss 1073741823
user_ID soft core 1073741823
user_ID hard core 1073741823
```

```

@perf soft nofile 1048576
@perf hard nofile 1048576
@perf soft fsize unlimited
@perf hard fsize unlimited
@perf soft data unlimited
@perf hard data unlimited
@perf soft stack unlimited
@perf hard stack unlimited
@perf soft rss unlimited
@perf hard rss unlimited
@perf soft core unlimited
@perf hard core unlimited
@perf soft cpu unlimited
@perf hard cpu unlimited
@perf hard as unlimited
@perf soft as unlimited
@perf soft locks unlimited
@perf hard locks unlimited
@perf soft memlock unlimited
@perf hard memlock unlimited
@perf soft memlock unlimited
@perf hard memlock unlimited

$ sudo gedit /etc/security/limits.conf
$ sudo gedit /etc/sysctl.conf
# Add following:
fs.file-max = 2097152
# Run:

sysctl -p

```

Additional References

- Load balance <https://github.com/google/seesaw>
- Go between balance <http://gobetween.io/downloads.html>
- Nginx balance <https://upcloud.com/community/tutorials/configure-load-balancing-nginx/>

Packages

Latex

```
jtarango@Telemetry-Bench-0:~$ sudo apt-get install texlive-pictures texlive-science texlive-latex-extra imagemagick -y
```

Bulk install

```
jtarango@Telemetry-Bench-0:~$ sudo apt-get update  
jtarango@Telemetry-Bench-0:~$ dpkg --clear-selections  
jtarango@Telemetry-Bench-0:~$ sudo dpkg --set-selections < list.txt
```

File: *list.txt* with file content below

accountsservice	install
acl	install
acpi-support	install
acpid	install
adduser	install
adium-theme-ubuntu	install
adwaita-icon-theme	install
aisleriot	install
alsa-base	install
alsa-utils	install
anacron	install
apg	install
app-install-data-partner	install
apparmor	install
apport	install
apport-gtk	install
apport-symptoms	install
appstream	install
apt	install
apt-config-icons	install
apt-utils	install
aptdaemon	install
aptdaemon-data	install
aptitude	install
aptitude-common	install
apturl	install
apturl-common	install

```
aspell                      install
aspell-en                   install
at                          install
at-spi2-core                install
autoconf                     install
automake                     install
autotools-dev                install
avahi-autoipd                install
avahi-daemon                 install
avahi-utils                  install
baobab                       install
base-files                   install
base-passwd                  install
bash                         install
bash-completion               install
bc                           install
bind9-host                   install
binutils                      install
binutils-common:amd64          install
binutils-x86-64-linux-gnu       install
bison                        install
blt                          install
bluez                        install
bluez-cups                   install
bluez-obexd                  install
bolt                         install
branding-ubuntu               install
brltty                       install
bsdmainutils                  install
bsdutils                      install
bubblewrap                    install
build-essential                install
busybox-initramfs              install
busybox-static                 install
byobu                         install
bzip2                        install
bzip2-doc                     install
ca-certificates                install
ccache                        install
cheese                        install
cheese-common                 install
chromium-codecs-ffmpeg-extra      install
cifs-utils                    install
```

```
cmake                                install
cmake-data                            install
colord                                install
colord-data                           install
command-not-found                     install
command-not-found-data                install
console-setup                         install
console-setup-linux                    install
coreutils                             install
cpio                                  install
cpp                                    install
cpp-7                                 install
cracklib-runtime                      install
crda                                  install
cron                                  install
cups                                  install
cups-browsed                          install
cups-bsd                              install
cups-client                           install
cups-common                           install
cups-core-drivers                     install
cups-daemon                           install
cups-filters                          install
cups-filters-core-drivers            install
cups-ipp-utils                         install
cups-pk-helper                        install
cups-ppdc                             install
cups-server-common                    install
dash                                  install
dbus                                  install
dbus-user-session                     install
dbus-x11                             install
dc                                     install
dconf-cli                            install
dconf-gsettings-backend:amd64        install
dconf-service                         install
dctrl-tools                           install
debconf                               install
debconf-i18n                           install
debianutils                           install
deja-dup                               install
desktop-file-utils                   install
device-tree-compiler                  install
```

devscripts	install
dfu-util	install
dh-python	install
dictionaries-common	install
diffstat	install
diffutils	install
dirmngr	install
distro-info-data	install
dmeventd	install
dmidecode	install
dmsetup	install
dmz-cursor-theme	install
dns-root-data	install
dnsmasq-base	install
dnsutils	install
docbook-xml	install
dosfstools	install
doxygen	install
dpkg	install
dpkg-dev	install
dput	install
e2fsprogs	install
ed	install
efibootmgr	install
eject	install
emacsen-common	install
enchant	install
eog	install
espeak-ng-data:amd64	install
evince	install
evince-common	install
evolution-data-server	install
evolution-data-server-common	install
example-content	install
fail2ban	install
fakeroot	install
fdisk	install
file	install
file-roller	install
findutils	install
fio	install
firefox	install
firefox-locale-en	install

flex	install
fontconfig	install
fontconfig-config	install
fonts-beng	install
fonts-beng-extra	install
fonts-dejavu-core	install
fonts-deva	install
fonts-deva-extra	install
fonts-droid-fallback	install
fonts-freefont-ttf	install
fonts-gargi	install
fonts-gubbi	install
fonts-gujr	install
fonts-gujr-extra	install
fonts-guru	install
fonts-guru-extra	install
fonts-indic	install
fonts-kacst	install
fonts-kacst-one	install
fonts-kalapi	install
fonts-khmeros-core	install
fonts-knda	install
fonts-lao	install
fonts-liberation	install
fonts-liberation2	install
fonts-lklug-sinhala	install
fonts-lohit-beng-assamese	install
fonts-lohit-beng-bengali	install
fonts-lohit-deva	install
fonts-lohit-gujr	install
fonts-lohit-guru	install
fonts-lohit-knda	install
fonts-lohit-mlym	install
fonts-lohit-orya	install
fonts-lohit-taml	install
fonts-lohit-taml-classical	install
fonts-lohit-telu	install
fonts-lyx	install
fonts-mlym	install
fonts-nakula	install
fonts-navilu	install
fonts-noto-cjk	install
fonts-noto-color-emoji	install

fonts-noto-mono	install
fonts-opensymbol	install
fonts-orya	install
fonts-orya-extra	install
fonts-pagul	install
fonts-sahadeva	install
fonts-samyak-deva	install
fonts-samyak-gujr	install
fonts-samyak-mlym	install
fonts-samyak-taml	install
fonts-sarai	install
fonts-sil-abyssinica	install
fonts-sil-padauk	install
fonts-smc	install
fonts-smc-anjalioldlipi	install
fonts-smc-chilanka	install
fonts-smc-dyuthi	install
fonts-smc-karumbi	install
fonts-smc-keraleeyam	install
fonts-smc-manjari	install
fonts-smc-meera	install
fonts-smc-rachana	install
fonts-smc-raghmalayalam	install
fonts-smc-suruma	install
fonts-smc-urop	install
fonts-taml	install
fonts-telu	install
fonts-telu-extra	install
fonts-thai-tlwg	install
fonts-tibetan-machine	install
fonts-tlwg-garuda	install
fonts-tlwg-garuda-ttf	install
fonts-tlwg-kinnari	install
fonts-tlwg-kinnari-ttf	install
fonts-tlwg-laksaman	install
fonts-tlwg-laksaman-ttf	install
fonts-tlwg-loma	install
fonts-tlwg-loma-ttf	install
fonts-tlwg-mono	install
fonts-tlwg-mono-ttf	install
fonts-tlwg-norasi	install
fonts-tlwg-norasi-ttf	install
fonts-tlwg-purisa	install

fonts-tlwg-purisa-ttf	install
fonts-tlwg-sawasdee	install
fonts-tlwg-sawasdee-ttf	install
fonts-tlwg-typewriter	install
fonts-tlwg-typewriter-ttf	install
fonts-tlwg-typist	install
fonts-tlwg-typist-ttf	install
fonts-tlwg-typo	install
fonts-tlwg-typo-ttf	install
fonts-tlwg-umpush	install
fonts-tlwg-umpush-ttf	install
fonts-tlwg-waree	install
fonts-tlwg-waree-ttf	install
fonts-ubuntu	install
foomatic-db-compressed-ppds	install
friendly-recovery	install
ftp	install
fuse	install
fwupd	install
fwupdate	install
fwupdate-signed	install
g++	install
g++-7	install
gawk	install
gcc	install
gcc-7	install
gcc-7-base:amd64	install
gcc-7-multilib	install
gcc-8-base:amd64	install
gcc-multilib	install
gcr	install
gdb	install
gdbserver	install
gdisk	install
gdm3	install
gedit	install
gedit-common	install
genisoimage	install
geoclue-2.0	install
geoip-database	install
gettext	install
gettext-base	install
ghostscript	install

```
ghostscript-x                                install
gir1.2-accountsservice-1.0                  install
gir1.2-atk-1.0:amd64                      install
gir1.2-atspi-2.0:amd64                    install
gir1.2-dbusmenu-glib-0.4:amd64            install
gir1.2-dee-1.0                            install
gir1.2-freedesktop:amd64                 install
gir1.2-gck-1:amd64                       install
gir1.2-gcr-3:amd64                      install
gir1.2-gdesktoopenums-3.0:amd64          install
gir1.2-gdkpixbuf-2.0:amd64              install
gir1.2-gdm-1.0                           install
gir1.2-geoclue-2.0:amd64                install
gir1.2-geocodelib-1.0:amd64             install
gir1.2-glib-2.0:amd64                   install
gir1.2-gmenu-3.0:amd64                 install
gir1.2-gnomebluetooth-1.0:amd64        install
gir1.2-gnomedesktop-3.0:amd64          install
gir1.2-goa-1.0:amd64                     install
gir1.2-gst-plugins-base-1.0:amd64       install
gir1.2-gstreamer-1.0:amd64              install
gir1.2-gtk-3.0:amd64                   install
gir1.2-gtksource-3.0:amd64            install
gir1.2-gudev-1.0:amd64                 install
gir1.2-gweather-3.0:amd64              install
gir1.2-harfbuzz-0.0:amd64              install
gir1.2-ibus-1.0:amd64                  install
gir1.2-javascriptcoregtk-4.0:amd64    install
gir1.2-json-1.0:amd64                  install
gir1.2-mutter-2:amd64                  install
gir1.2-nm-1.0:amd64                   install
gir1.2-nma-1.0:amd64                  install
gir1.2-notify-0.7:amd64               install
gir1.2-packagekitglib-1.0            install
gir1.2-pango-1.0:amd64                install
gir1.2-peas-1.0:amd64                install
gir1.2-polkit-1.0                     install
gir1.2-rb-3.0:amd64                   install
gir1.2-rsvg-2.0:amd64                install
gir1.2-secret-1:amd64                install
gir1.2-snapd-1:amd64                 install
gir1.2-soup-2.4:amd64                install
gir1.2-totem-1.0:amd64              install
```

gir1.2-totemparser-1.0:amd64	install
gir1.2-udisks-2.0:amd64	install
gir1.2-unity-5.0:amd64	install
gir1.2-upowerglib-1.0:amd64	install
gir1.2-vte-2.91:amd64	install
gir1.2-webkit2-4.0:amd64	install
gir1.2-wnck-3.0:amd64	install
git	install
git-man	install
gjs	install
gkbd-capplet	install
glances	install
glib-networking:amd64	install
glib-networking-common	install
glib-networking-services	install
gnome-accessibility-themes	install
gnome-bluetooth	install
gnome-calendar	install
gnome-control-center	install
gnome-control-center-data	install
gnome-control-center-faces	install
gnome-desktop3-data	install
gnome-disk-utility	install
gnome-font-viewer	install
gnome-getting-started-docs	install
gnome-initial-setup	install
gnome-keyring	install
gnome-keyring-pkcs11:amd64	install
gnome-mahjongg	install
gnome-menus	install
gnome-mines	install
gnome-online-accounts	install
gnome-power-manager	install
gnome-screenshot	install
gnome-session-bin	install
gnome-session-canberra	install
gnome-session-common	install
gnome-settings-daemon	install
gnome-settings-daemon-schemas	install
gnome-shell	install
gnome-shell-common	install
gnome-shell-extension-appindicator	install
gnome-shell-extension-ubuntu-dock	install

gnome-software	install
gnome-software-common	install
gnome-software-plugin-snap	install
gnome-startup-applications	install
gnome-sudoku	install
gnome-terminal	install
gnome-terminal-data	install
gnome-themes-extra:amd64	install
gnome-themes-extra-data	install
gnome-todo	install
gnome-todo-common	install
gnome-user-docs	install
gnome-user-guide	install
gnome-video-effects	install
gnupg	install
gnupg-110n	install
gnupg-utils	install
golang-docker-credential-helpers	install
gperf	install
gpg	install
gpg-agent	install
gpg-wks-client	install
gpg-wks-server	install
gpgconf	install
gpgsm	install
gpgv	install
grep	install
grilo-plugins-0.3-base:amd64	install
groff-base	install
grub-common	install
grub-efi-amd64-bin	install
grub-efi-amd64-signed	install
grub-gfxpayload-lists	install
grub-pc	install
grub-pc-bin	install
grub2-common	install
gsettings-desktop-schemas	install
gsettings-ubuntu-schemas	install
gsfonts	install
gstreamer1.0-alsa:amd64	install
gstreamer1.0-clutter-3.0:amd64	install
gstreamer1.0-fluendo-mp3:amd64	install
gstreamer1.0-gl:amd64	install

gstreamer1.0-gtk3:amd64	install
gstreamer1.0-libav:amd64	install
gstreamer1.0-packagekit	install
gstreamer1.0-plugins-base:amd64	install
gstreamer1.0-plugins-base-apps	install
gstreamer1.0-plugins-good:amd64	install
gstreamer1.0-plugins-ugly:amd64	install
gstreamer1.0-pulseaudio:amd64	install
gstreamer1.0-tools	install
gstreamer1.0-vaapi:amd64	install
gstreamer1.0-x:amd64	install
gtk-update-icon-cache	install
gtk2-engines-murrine:amd64	install
gtk2-engines-pixbuf:amd64	install
guile-2.0-libs:amd64	install
gvfs:amd64	install
gvfs-backends	install
gvfs-bin	install
gvfs-common	install
gvfs-daemons	install
gvfs-fuse	install
gvfs-libs:amd64	install
gzip	install
hddtemp	install
hdparm	install
hicolor-icon-theme	install
hostname	install
hplip	install
hplip-data	install
htop	install
humanity-icon-theme	install
hunspell-en-us	install
hyphen-en-us	install
i965-va-driver:amd64	install
ibus	install
ibus-gtk:amd64	install
ibus-gtk3:amd64	install
ibus-table	install
ibverbs-providers:amd64	install
icu-devtools	install
ifupdown	install
io-sensor-proxy	install
im-config	install

imagemagick	install
imagemagick-6-common	install
imagemagick-6.q16	install
info	install
init	install
init-system-helpers	install
initramfs-tools	install
initramfs-tools-bin	install
initramfs-tools-core	install
inputattach	install
install-info	install
intltool-debian	install
ippusbxd	install
iproute2	install
iptables	install
iputils-arping	install
iputils-ping	install
iputils-tracepath	install
irqbalance	install
isc-dhcp-client	install
isc-dhcp-common	install
isdct	install
iso-codes	install
iw	install
javascript-common	install
kbd	install
kerneloops	install
keyboard-configuration	install
klibc-utils	install
kmmod	install
krb5-locales	install
language-pack-en	install
language-pack-en-base	install
language-pack-gnome-en	install
language-pack-gnome-en-base	install
language-selector-common	install
language-selector-gnome	install
laptop-detect	install
less	install
lib32asan4	install
lib32atomic1	install
lib32cilkrt5	install
lib32gcc-7-dev	install

lib32gcc1	install
lib32gomp1	install
lib32itm1	install
lib32mpx2	install
lib32quadmath0	install
lib32stdc++6	install
lib32ubsan0	install
liba52-0.7.4:amd64	install
libaal:amd64	install
libaacs0:amd64	install
libabw-0.1-1:amd64	install
libaccountsservice0:amd64	install
libacl1:amd64	install
libaio-dev:amd64	install
libaiol:amd64	install
libalgorithm-diff-perl	install
libalgorithm-diff-xs-perl	install
libalgorithm-merge-perl	install
libao-common	install
libao4:amd64	install
libapparmor1:amd64	install
libappindicator3-1	install
libappstream-glib8:amd64	install
libappstream4:amd64	install
libapt-inst2.0:amd64	install
libapt-pkg-perl	install
libapt-pkg5.0:amd64	install
libarchive-zip-perl	install
libarchive13:amd64	install
libargon2-0:amd64	install
libart-2.0-2:amd64	install
libasan4:amd64	install
libasn1-8-heimdal:amd64	install
libasound2:amd64	install
libasound2-data	install
libasound2-plugins:amd64	install
libaspell15:amd64	install
libass9:amd64	install
libassuan0:amd64	install
libasyncns0:amd64	install
libatasmart4:amd64	install
libatk-adaptor:amd64	install
libatk-bridge2.0-0:amd64	install

libatk1.0-0:amd64	install
libatk1.0-data	install
libatm1:amd64	install
libatomic1:amd64	install
libatspi2.0-0:amd64	install
libattr1:amd64	install
libaudio2:amd64	install
libaudit-common	install
libaudit1:amd64	install
libauthen-sasl-perl	install
libavahi-client3:amd64	install
libavahi-common-data:amd64	install
libavahi-common3:amd64	install
libavahi-core7:amd64	install
libavahi-glib1:amd64	install
libavahi-ui-gtk3-0:amd64	install
libavc1394-0:amd64	install
libavcodec57:amd64	install
libavfilter6:amd64	install
libavformat57:amd64	install
libavresample3:amd64	install
libavutil55:amd64	install
libb-hooks-endofscope-perl	install
libb-hooks-op-check-perl	install
libbabeltrace1:amd64	install
libbdplus0:amd64	install
libbind9-160:amd64	install
libbinutils:amd64	install
libbison-dev:amd64	install
libblas3:amd64	install
libblkid1:amd64	install
libblockdev-crypto2:amd64	install
libblockdev-fs2:amd64	install
libblockdev-loop2:amd64	install
libblockdev-part-err2:amd64	install
libblockdev-part2:amd64	install
libblockdev-swap2:amd64	install
libblockdev-utils2:amd64	install
libblockdev2:amd64	install
libbluetooth3:amd64	install
libbluray2:amd64	install
libboost-all-dev	install
libboost-atomic-dev:amd64	install

```
libboost-atomic1.65-dev:amd64           install
libboost-atomic1.65.1:amd64             install
libboost-chrono-dev:amd64              install
libboost-chrono1.65-dev:amd64          install
libboost-chrono1.65.1:amd64            install
libboost-container-dev:amd64           install
libboost-container1.65-dev:amd64       install
libboost-container1.65.1:amd64         install
libboost-context-dev:amd64             install
libboost-context1.65-dev:amd64         install
libboost-context1.65.1:amd64           install
libboost-coroutine-dev:amd64           install
libboost-coroutine1.65-dev:amd64       install
libboost-coroutine1.65.1:amd64         install
libboost-date-time-dev:amd64           install
libboost-date-time1.65-dev:amd64       install
libboost-date-time1.65.1:amd64         install
libboost-dev:amd64                     install
libboost-exception-dev:amd64           install
libboost-exception1.65-dev:amd64        install
libboost-fiber-dev:amd64               install
libboost-fiber1.65-dev:amd64           install
libboost-fiber1.65.1:amd64             install
libboost-filesystem-dev:amd64          install
libboost-filesystem1.65-dev:amd64       install
libboost-filesystem1.65.1:amd64         install
libboost-graph-dev:amd64               install
libboost-graph-parallel-dev           install
libboost-graph-parallel1.65-dev        install
libboost-graph-parallel1.65.1          install
libboost-graph1.65-dev:amd64           install
libboost-graph1.65.1:amd64             install
libboost-iostreams-dev:amd64           install
libboost-iostreams1.65-dev:amd64        install
libboost-iostreams1.65.1:amd64         install
libboost-locale-dev:amd64              install
libboost-locale1.65-dev:amd64           install
libboost-locale1.65.1:amd64             install
libboost-log-dev                      install
libboost-log1.65-dev                  install
libboost-log1.65.1                    install
libboost-math-dev:amd64                install
libboost-math1.65-dev:amd64             install
```

libboost-math1.65.1:amd64	install
libboost-mpi-dev	install
libboost-mpi-python-dev	install
libboost-mpi-python1.65-dev	install
libboost-mpi-python1.65.1	install
libboost-mpi1.65-dev	install
libboost-mpi1.65.1	install
libboost-numpy-dev	install
libboost-numpy1.65-dev	install
libboost-numpy1.65.1	install
libboost-program-options-dev:amd64	install
libboost-program-options1.65-dev:amd64	install
libboost-program-options1.65.1:amd64	install
libboost-python-dev	install
libboost-python1.65-dev	install
libboost-python1.65.1	install
libboost-random-dev:amd64	install
libboost-random1.65-dev:amd64	install
libboost-random1.65.1:amd64	install
libboost-regex-dev:amd64	install
libboost-regex1.65-dev:amd64	install
libboost-regex1.65.1:amd64	install
libboost-serialization-dev:amd64	install
libboost-serialization1.65-dev:amd64	install
libboost-serialization1.65.1:amd64	install
libboost-signals-dev:amd64	install
libboost-signals1.65-dev:amd64	install
libboost-signals1.65.1:amd64	install
libboost-stacktrace-dev:amd64	install
libboost-stacktrace1.65-dev:amd64	install
libboost-stacktrace1.65.1:amd64	install
libboost-system-dev:amd64	install
libboost-system1.65-dev:amd64	install
libboost-system1.65.1:amd64	install
libboost-test-dev:amd64	install
libboost-test1.65-dev:amd64	install
libboost-test1.65.1:amd64	install
libboost-thread-dev:amd64	install
libboost-thread1.65-dev:amd64	install
libboost-thread1.65.1:amd64	install
libboost-timer-dev:amd64	install
libboost-timer1.65-dev:amd64	install
libboost-timer1.65.1:amd64	install

```
libboost-tools-dev                                install
libboost-type-erasure-dev:amd64                  install
libboost-type-erasure1.65-dev:amd64              install
libboost-type-erasure1.65.1:amd64                install
libboost-wave-dev:amd64                         install
libboost-wave1.65-dev:amd64                     install
libboost-wave1.65.1:amd64                      install
libboost1.65-dev:amd64                         install
libboost1.65-tools-dev                          install
libbrlapi0.6:amd64                            install
libbrotli1:amd64                             install
libbs2b0:amd64                                install
libbsd0:amd64                                 install
libbz2-1.0:amd64                             install
libbz2-dev:amd64                            install
libc-bin                                       install
libc-dev-bin                                  install
libc6:amd64                                    install
libc6-dbg:amd64                             install
libc6-dev:amd64                            install
libc6-dev-i386                               install
libc6-dev-x32                                install
libc6-i386                                    install
libc6-x32                                     install
libcaca0:amd64                               install
libcairo-gobject-perl                        install
libcairo-gobject2:amd64                       install
libcairo-perl                                install
libcairo2:amd64                               install
libcamel-1.2-61:amd64                        install
libcanberra-gtk3-0:amd64                      install
libcanberra-gtk3-module:amd64                 install
libcanberra-pulse:amd64                       install
libcanberra0:amd64                           install
libcap-ng0:amd64                            install
libcap2:amd64                                install
libcap2-bin                                   install
libcc1-0:amd64                               install
libcdio-cdda2:amd64                          install
libcdio-paranoia2:amd64                      install
libcdio17:amd64                             install
libcdparanoia0:amd64                         install
libcdr-0.1-1:amd64                           install
```

libcgi-fast-perl	install
libcgi-pm-perl	install
libcheese-gtk25:amd64	install
libcheese8:amd64	install
libchromaprint1:amd64	install
libcilkrt5:amd64	install
libclang1-6.0:amd64	install
libclass-accessor-perl	install
libclass-method-modifiers-perl	install
libclass-xsaccessor-perl	install
libclone-perl	install
libclucene-contribs1v5:amd64	install
libclucene-core1v5:amd64	install
libclutter-1.0-0:amd64	install
libclutter-1.0-common	install
libclutter-gst-3.0-0:amd64	install
libclutter-gtk-1.0-0:amd64	install
libcmis-0.5-5v5	install
libcogl-common	install
libcogl-pango20:amd64	install
libcogl-path20:amd64	install
libcogl20:amd64	install
libcolamd2:amd64	install
libcolord-gtk1:amd64	install
libcolord2:amd64	install
libcolorhug2:amd64	install
libcom-err2:amd64	install
libcrack2:amd64	install
libcroco3:amd64	install
libcryptsetup12:amd64	install
libcrystalhd3:amd64	install
libcups2:amd64	install
libcupscgil1:amd64	install
libcupsfilters1:amd64	install
libcupsimage2:amd64	install
libcupsmime1:amd64	install
libcupsppdc1:amd64	install
libcurl3-gnutls:amd64	install
libcurl4:amd64	install
libcwidget3v5:amd64	install
libdaemon0:amd64	install
libdata-dump-perl	install
libdata-optlist-perl	install

libdatriel:amd64	install
libdazzle-1.0-0:amd64	install
libdb5.3:amd64	install
libdbus-1-3:amd64	install
libdbus-glib-1-2:amd64	install
libdbusmenu-glib4:amd64	install
libdbusmenu-gtk3-4:amd64	install
libdbusmenu-gtk4:amd64	install
libdconf1:amd64	install
libdebconfclient0:amd64	install
libdee-1.0-4:amd64	install
libdevel-callchecker-perl	install
libdevel-globaldestruction-perl	install
libdevmapper-event1.02.1:amd64	install
libdevmapper1.02.1:amd64	install
libdigest-hmac-perl	install
libdistro-info-perl	install
libdjvulibre-text	install
libdjvulibre21:amd64	install
libdmapsharing-3.0-2:amd64	install
libdns-export1100	install
libdns1100:amd64	install
libdotconf0:amd64	install
libdpkg-perl	install
libdrm-amdgpu1:amd64	install
libdrm-common	install
libdrm-intel1:amd64	install
libdrm-nouveau2:amd64	install
libdrm-radeon1:amd64	install
libdrm2:amd64	install
libdv4:amd64	install
libdvdnav4:amd64	install
libdvdread4:amd64	install
libdw1:amd64	install
libdynaloader-functions-perl	install
libe-book-0.1-1:amd64	install
libebbackend-1.2-10:amd64	install
libebook-1.2-19:amd64	install
libebook-contacts-1.2-2:amd64	install
libecal-1.2-19:amd64	install
libedata-book-1.2-25:amd64	install
libedata-cal-1.2-28:amd64	install
libedataserver-1.2-23:amd64	install

libedataserverui-1.2-2:amd64	install
libedit2:amd64	install
libefiboot1:amd64	install
libefivar1:amd64	install
libegl-mesa0:amd64	install
libegl1:amd64	install
libegl1-mesa:amd64	install
libelf1:amd64	install
libemail-valid-perl	install
libenchant1c2a:amd64	install
libencode-locale-perl	install
libeot0:amd64	install
libepoxy0:amd64	install
libept1.5.0:amd64	install
libepubgen-0.1-1:amd64	install
liberror-perl	install
libespeak-ng1:amd64	install
libestr0:amd64	install
libetonyek-0.1-1:amd64	install
libevdev2:amd64	install
libevdocument3-4:amd64	install
libevent-2.1-6:amd64	install
libevview3-3:amd64	install
libexemp13:amd64	install
libexif12:amd64	install
libexiv2-14:amd64	install
libexpat1:amd64	install
libexpat1-dev:amd64	install
libexporter-tiny-perl	install
libext2fs2:amd64	install
libexttextcat-2.0-0:amd64	install
libexttextcat-data	install
libfabric1	install
libfakeroot:amd64	install
libfastjson4:amd64	install
libfcgi-perl	install
libfdisk1:amd64	install
libffi6:amd64	install
libfftw3-double3:amd64	install
libfftw3-single3:amd64	install
libfile-basedir-perl	install
libfile-chdir-perl	install
libfile-copy-recursive-perl	install

libfile-desktopentry-perl	install
libfile-fcntllock-perl	install
libfile-homedir-perl	install
libfile-listing-perl	install
libfile-mimeinfo-perl	install
libfile-which-perl	install
libfl-dev:amd64	install
libfl2:amd64	install
libflac8:amd64	install
libflitel1:amd64	install
libfont-afm-perl	install
libfontconfig1:amd64	install
libfontembed1:amd64	install
libfontenc1:amd64	install
libfreehand-0.1-1	install
libfreerdp-client2-2:amd64	install
libfreerdp2-2:amd64	install
libfreetype6:amd64	install
libfribidi0:amd64	install
libfuse2:amd64	install
libfwupd1:amd64	install
libfwupd2:amd64	install
libgail-3-0:amd64	install
libgail-common:amd64	install
libgail18:amd64	install
libgbm1:amd64	install
libgclc2:amd64	install
libgcab-1.0-0:amd64	install
libgcc-7-dev:amd64	install
libgcc1:amd64	install
libgck-1-0:amd64	install
libgcr-base-3-1:amd64	install
libgcr-ui-3-1:amd64	install
libgcrypt20:amd64	install
libgd3:amd64	install
libgdata-common	install
libgdata22:amd64	install
libgdbm-compat4:amd64	install
libgdbm5:amd64	install
libgdk-pixbuf2.0-0:amd64	install
libgdk-pixbuf2.0-bin	install
libgdk-pixbuf2.0-common	install
libgdm1	install

libgee-0.8-2:amd64	install
libgeoclue-2-0:amd64	install
libgeocode-glib0:amd64	install
libgeoip1:amd64	install
libgetopt-long-descriptive-perl	install
libgexiv2-2:amd64	install
libgfotran4:amd64	install
libgirepository-1.0-1:amd64	install
libgit-wrapper-perl	install
libgjs0g	install
libgl1:amd64	install
libgl1-mesa-dri:amd64	install
libgl1-mesa-glx:amd64	install
libglapi-mesa:amd64	install
libgles2:amd64	install
libglib-object-introspection-perl	install
libglib-perl	install
libglib2.0-0:amd64	install
libglib2.0-bin	install
libglib2.0-data	install
libglib2.0-dev:amd64	install
libglib2.0-dev-bin	install
libglu1-mesa:amd64	install
libglvnd0:amd64	install
libglx-mesa0:amd64	install
libglx0:amd64	install
libgme0:amd64	install
libgmime-3.0-0:amd64	install
libgmp10:amd64	install
libgnome-autoar-0-0:amd64	install
libgnome-bluetooth13:amd64	install
libgnome-desktop-3-17:amd64	install
libgnome-games-support-1-3:amd64	install
libgnome-games-support-common	install
libgnome-menu-3-0:amd64	install
libgnome-todo	install
libgnomekbd-common	install
libgnomekbd8:amd64	install
libgnutls30:amd64	install
libgoa-1.0-0b:amd64	install
libgoa-1.0-common	install
libgoa-backend-1.0-1:amd64	install
libgom-1.0-0:amd64	install

libgomp1:amd64	install
libgpg-error0:amd64	install
libgpgme11:amd64	install
libgpgmep6:amd64	install
libgphoto2-6:amd64	install
libgphoto2-110n	install
libgphoto2-port12:amd64	install
libgpm2:amd64	install
libgpod-common	install
libgpod4:amd64	install
libgraphene-1.0-0:amd64	install
libgraphite2-3:amd64	install
libgraphite2-dev:amd64	install
libgrilo-0.3-0:amd64	install
libgs9:amd64	install
libgs9-common	install
libgsml1:amd64	install
libgspell-1-1:amd64	install
libgspell-1-common	install
libgssapi-krb5-2:amd64	install
libgssapi3-heimdal:amd64	install
libgstreamer-gl1.0-0:amd64	install
libgstreamer-plugins-bad1.0-0:amd64	install
libgstreamer-plugins-base1.0-0:amd64	install
libgstreamer-plugins-good1.0-0:amd64	install
libgstreamer1.0-0:amd64	install
libgtk-3-0:amd64	install
libgtk-3-bin	install
libgtk-3-common	install
libgtk2-perl	install
libgtk2.0-0:amd64	install
libgtk2.0-bin	install
libgtk2.0-common	install
libgtk3-perl	install
libgtksourceview-3.0-1:amd64	install
libgtksourceview-3.0-common	install
libgtop-2.0-11:amd64	install
libgtop2-common	install
libgudev-1.0-0:amd64	install
libgusb2:amd64	install
libgutenprint2	install
libgweather-3-15:amd64	install
libgweather-common	install

libgxpath2:amd64	install
libharfbuzz-dev:amd64	install
libharfbuzz-gobject0:amd64	install
libharfbuzz-icu0:amd64	install
libharfbuzz0b:amd64	install
libhcrypto4-heimdal:amd64	install
libheimbase1-heimdal:amd64	install
libheimntlm0-heimdal:amd64	install
libhogweed4:amd64	install
libhpmd0:amd64	install
libhtml-form-perl	install
libhtml-format-perl	install
libhtml-parser-perl	install
libhtml-tagset-perl	install
libhtml-tree-perl	install
libhttp-cookies-perl	install
libhttp-daemon-perl	install
libhttp-date-perl	install
libhttp-message-perl	install
libhttp-negotiate-perl	install
libhunspell-1.6-0:amd64	install
libhwloc-dev:amd64	install
libhwloc-plugins	install
libhwloc5:amd64	install
libhx509-5-heimdal:amd64	install
libhyphen0:amd64	install
libibus-1.0-5:amd64	install
libibverbs-dev:amd64	install
libibverbs1:amd64	install
libical3:amd64	install
libice6:amd64	install
libicu-dev	install
libicu-le-hb-dev:amd64	install
libicu-le-hb0:amd64	install
libicu60:amd64	install
libiculx60:amd64	install
libidn11:amd64	install
libidn2-0:amd64	install
libiec61883-0:amd64	install
libieee1284-3:amd64	install
libijs-0.35:amd64	install
libilmbase12:amd64	install
libimobiledevice6:amd64	install

libimport-into-perl	install
libindicator3-7	install
libinput-bin	install
libinput10:amd64	install
libio-html-perl	install
libio-pty-perl	install
libio-socket-inet6-perl	install
libio-socket-ssl-perl	install
libio-string-perl	install
libio-stringy-perl	install
libip4tc0:amd64	install
libip6tc0:amd64	install
libipc-run-perl	install
libipc-system-simple-perl	install
libiptc0:amd64	install
libirs160:amd64	install
libisc-export169:amd64	install
libisc169:amd64	install
libisccc160:amd64	install
libisccfg160:amd64	install
libisl19:amd64	install
libitm1:amd64	install
libiw30:amd64	install
libjack-jackd2-0:amd64	install
libjansson4:amd64	install
libjavascriptcoregtk-4.0-18:amd64	install
libjbig0:amd64	install
libjbig2dec0:amd64	install
libjpeg-turbo8:amd64	install
libjpeg8:amd64	install
libjs-jquery	install
libjs-jquery-ui	install
libjson-c3:amd64	install
libjson-glib-1.0-0:amd64	install
libjson-glib-1.0-common	install
libjsoncpp1:amd64	install
libk5crypto3:amd64	install
libkeyutils1:amd64	install
libklibc	install
libkmod2:amd64	install
libkpathsea6:amd64	install
libkrb5-26-heimdal:amd64	install
libkrb5-3:amd64	install

libkrb5support0:amd64	install
libksba8:amd64	install
liblangtag-common	install
liblangtag1:amd64	install
liblapack3:amd64	install
liblcms2-2:amd64	install
liblcms2-utils	install
libldap-2.4-2:amd64	install
libldap-common	install
libldb1:amd64	install
liblirc-client0:amd64	install
liblist-compare-perl	install
liblist-moreutils-perl	install
libllvm6.0:amd64	install
libllvm9:amd64	install
liblocale-gettext-perl	install
liblouis-data	install
liblouis14:amd64	install
liblouisutdml-bin	install
liblouisutdml-data	install
liblouisutdml8:amd64	install
liblqr-1-0:amd64	install
liblsan0:amd64	install
libltdl-dev:amd64	install
libltdl7:amd64	install
liblua5.3-0:amd64	install
liblvm2app2.2:amd64	install
liblvm2cmd2.02:amd64	install
liblwp-mediatypes-perl	install
liblwp-protocol-https-perl	install
liblwres160:amd64	install
liblz4-1:amd64	install
liblzma5:amd64	install
liblzo2-2:amd64	install
libmagic-mgc	install
libmagic1:amd64	install
libmagickcore-6.q16-3:amd64	install
libmagickcore-6.q16-3-extra:amd64	install
libmagickwand-6.q16-3:amd64	install
libmailtools-perl	install
libmbim-glib4:amd64	install
libmbim-proxy	install
libmediaart-2.0-0:amd64	install

libmessaging-menu0:amd64	install
libmhash2:amd64	install
libminiupnpc10:amd64	install
libmm-glib0:amd64	install
libmn10:amd64	install
libmodule-implementation-perl	install
libmodule-runtime-perl	install
libmoo-perl	install
libmount1:amd64	install
libmozjs-52-0:amd64	install
libmp3lame0:amd64	install
libmpc3:amd64	install
libmpdec2:amd64	install
libmpeg2-4:amd64	install
libmpfr6:amd64	install
libmpg123-0:amd64	install
libmpx2:amd64	install
libmspub-0.1-1:amd64	install
libmtdev1:amd64	install
libmtp-common	install
libmtp-runtime	install
libmtp9:amd64	install
libmutter-2-0:amd64	install
libmwaw-0.3-3:amd64	install
libmysofa0:amd64	install
libmythes-1.2-0:amd64	install
libnamespace-clean-perl	install
libnatpmp1	install
libnautilus-extension1a:amd64	install
libncurses5:amd64	install
libncurses5-dev:amd64	install
libncursesw5:amd64	install
libndp0:amd64	install
libneon27-gnutls:amd64	install
libnet-dbus-perl	install
libnet-dns-perl	install
libnet-domain-tld-perl	install
libnet-http-perl	install
libnet-ip-perl	install
libnet-libidn-perl	install
libnet-smtp-ssl-perl	install
libnet-ssleay-perl	install
libnetfilter-conntrack3:amd64	install

libnetpbm10	install
libnettle6:amd64	install
libnewt0.52:amd64	install
libnfsnetlink0:amd64	install
libnghtp2-14:amd64	install
libnih1:amd64	install
libnl-3-200:amd64	install
libnl-genl-3-200:amd64	install
libnl-route-3-200:amd64	install
libnm0:amd64	install
libnma0:amd64	install
libnorm1:amd64	install
libnotify-bin	install
libnotify4:amd64	install
libnpth0:amd64	install
libnspr4:amd64	install
libnss-mdns:amd64	install
libnss-myhostname:amd64	install
libnss-systemd:amd64	install
libnss3:amd64	install
libntfs-3g88	install
libnuma-dev:amd64	install
libnuma1:amd64	install
libnumber-compare-perl	install
libnumber-range-perl	install
liboauth0:amd64	install
libodfgen-0.1-1:amd64	install
libogg0:amd64	install
libopencore-amrnb0:amd64	install
libopencore-amrwb0:amd64	install
libopenexr22:amd64	install
libopenjp2-7:amd64	install
libopenmpi-dev	install
libopenmpi2:amd64	install
libopenmpt0:amd64	install
libopts25:amd64	install
libopus0:amd64	install
liborc-0.4-0:amd64	install
liborcus-0.13-0:amd64	install
libp11-kit0:amd64	install
libpackage-stash-perl	install
libpackage-stash-xs-perl	install
libpackagekit-glib2-18:amd64	install

libpagemaker-0.0-0:amd64	install
libpam-cap:amd64	install
libpam-gnome-keyring:amd64	install
libpam-modules:amd64	install
libpam-modules-bin	install
libpam-runtime	install
libpam-systemd:amd64	install
libpam0g:amd64	install
libpango-1.0-0:amd64	install
libpango-perl	install
libpangocairo-1.0-0:amd64	install
libpangoft2-1.0-0:amd64	install
libpangoxft-1.0-0:amd64	install
libpaper-utils	install
libpaper1:amd64	install
libparams-classify-perl	install
libparams-util-perl	install
libparams-validate-perl	install
libparse-debianchangelog-perl	install
libparted-fs-resize0:amd64	install
libparted2:amd64	install
libpath-iterator-rule-perl	install
libpath-tiny-perl	install
libpcap0.8:amd64	install
libpcaudio0	install
libpci3:amd64	install
libpciaccess0:amd64	install
libpcre16-3:amd64	install
libpcre3:amd64	install
libpcre3-dev:amd64	install
libpcre32-3:amd64	install
libpcrecpp0v5:amd64	install
libpcsclite1:amd64	install
libpeas-1.0-0:amd64	install
libpeas-common	install
libperl5.26:amd64	install
libperlio-gzip-perl	install
libpgm-5.2-0:amd64	install
libphonenumber7:amd64	install
libpipeline1:amd64	install
libpixman-1-0:amd64	install
libplist3:amd64	install
libplymouth4:amd64	install

libpng16-16:amd64	install
libpod-constants-perl	install
libpolkit-agent-1-0:amd64	install
libpolkit-backend-1-0:amd64	install
libpolkit-gobject-1-0:amd64	install
libpoppler-glib8:amd64	install
libpoppler73:amd64	install
libpopt0:amd64	install
libpostproc54:amd64	install
libprocps6:amd64	install
libprotobuf10:amd64	install
libproxy1-plugin-gsettings:amd64	install
libproxy1-plugin-networkmanager:amd64	install
libproxy1v5:amd64	install
libpsl5:amd64	install
libpsm-infinopath1	install
libpulse-mainloop-glib0:amd64	install
libpulse0:amd64	install
libpulsedsp:amd64	install
libpwquality-common	install
libpwquality1:amd64	install
libpython-all-dev:amd64	install
libpython-dev:amd64	install
libpython-stdlib:amd64	install
libpython2.7:amd64	install
libpython2.7-dev:amd64	install
libpython2.7-minimal:amd64	install
libpython2.7-stdlib:amd64	install
libpython3-dev:amd64	install
libpython3-stdlib:amd64	install
libpython3.6:amd64	install
libpython3.6-dev:amd64	install
libpython3.6-minimal:amd64	install
libpython3.6-stdlib:amd64	install
libqmi-glib5:amd64	install
libqmi-proxy	install
libqpdf21:amd64	install
libqqwing2v5:amd64	install
libquadmath0:amd64	install
librados2	install
libraptor2-0:amd64	install
librarian0	install
librasqal3:amd64	install

libraw1394-11:amd64	install
libraw16:amd64	install
librbd1	install
librdf0:amd64	install
librdmacm1:amd64	install
libreadline5:amd64	install
libreadline7:amd64	install
libregexp-pattern-license-perl	install
libreoffice-avmedia-backend-gstreamer	install
libreoffice-base-core	install
libreoffice-calc	install
libreoffice-common	install
libreoffice-core	install
libreoffice-draw	install
libreoffice-gnome	install
libreoffice-gtk3	install
libreoffice-help-en-us	install
libreoffice-impress	install
libreoffice-math	install
libreoffice-ogltrans	install
libreoffice-pdfimport	install
libreoffice-style-breeze	install
libreoffice-style-galaxy	install
libreoffice-style-tango	install
libreoffice-writer	install
librest-0.7-0:amd64	install
librevenge-0.0-0:amd64	install
librhash0:amd64	install
librhythmbox-core10:amd64	install
libroken18-heimdal:amd64	install
librole-tiny-perl	install
librsvg2-2:amd64	install
librsvg2-common:amd64	install
librtmp1:amd64	install
librubberband2:amd64	install
libsamplerate0:amd64	install
libsane-common	install
libsane-hpaio:amd64	install
libsane1:amd64	install
libsasl2-2:amd64	install
libsasl2-modules:amd64	install
libsasl2-modules-db:amd64	install
libsbc1:amd64	install

libseccomp2:amd64	install
libsecret-1-0:amd64	install
libsecret-common	install
libselinux1:amd64	install
libsemanage-common	install
libsemanage1:amd64	install
libsensors4:amd64	install
libsepoll:amd64	install
libsgutils2-2	install
libshine3:amd64	install
libshout3:amd64	install
libsidplay1v5:amd64	install
libsigc++-2.0-0v5:amd64	install
libsigsegv2:amd64	install
libslang2:amd64	install
libsm6:amd64	install
libsmartcols1:amd64	install
libsmbclient:amd64	install
libsmbios-c2	install
libsnapd-glib1:amd64	install
libsnappy1v5:amd64	install
libsndfile1:amd64	install
libsnmp-base	install
libsntp30:amd64	install
libsocket6-perl	install
libsodium23:amd64	install
libsonic0:amd64	install
libsort-key-perl	install
libsort-versions-perl	install
libsoup-gnome2.4-1:amd64	install
libsoup2.4-1:amd64	install
libsoxr0:amd64	install
libspectre1:amd64	install
libspeechd2:amd64	install
libspeex1:amd64	install
libspeexdsp1:amd64	install
libsqlite3-0:amd64	install
libss2:amd64	install
libssh-4:amd64	install
libssh-gcrypt-4:amd64	install
libssl-dev:amd64	install
libssl1.0.0:amd64	install
libssl1.1:amd64	install

libstartup-notification0:amd64	install
libstdc++-7-dev:amd64	install
libstdc++6:amd64	install
libstemmer0d:amd64	install
libstrictures-perl	install
libstring-copyright-perl	install
libstring-escape-perl	install
libsub-exporter-perl	install
libsub-exporter-progressive-perl	install
libsub-identify-perl	install
libsub-install-perl	install
libsub-name-perl	install
libsub-quote-perl	install
libsuitesparseconfig5:amd64	install
libswresample2:amd64	install
libswscale4:amd64	install
libsysmetrics1:amd64	install
libsystemd-dev:amd64	install
libsystemd0:amd64	install
libtag1v5:amd64	install
libtag1v5-vanilla:amd64	install
libtalloc2:amd64	install
libtasn1-6:amd64	install
libtcl8.6:amd64	install
libtdb1:amd64	install
libteamdctl0:amd64	install
libtevent0:amd64	install
libtext-charwidth-perl	install
libtext-glob-perl	install
libtext-iconv-perl	install
libtext-levenshtein-perl	install
libtext-wrapi18n-perl	install
libthai-data	install
libthai0:amd64	install
libtheora0:amd64	install
libtie-ixhash-perl	install
libtiff5:amd64	install
libtimedate-perl	install
libtinfo-dev:amd64	install
libtinfo5:amd64	install
libtk8.6:amd64	install
libtool	install
libtotem-plparser-common	install

libtotem-plparser18:amd64	install
libtotem0:amd64	install
libtracker-sparql-2.0-0:amd64	install
libtry-tiny-perl	install
libtsan0:amd64	install
libtwolame0:amd64	install
libu2f-udev	install
libubsan0:amd64	install
libudev1:amd64	install
libudisks2-0:amd64	install
libunicode-utf8-perl	install
libunistring2:amd64	install
libunity-protocol-private0:amd64	install
libunity-scopes-json-def-desktop	install
libunity9:amd64	install
libunwind8:amd64	install
libupower-glib3:amd64	install
liburi-perl	install
libusb-1.0-0:amd64	install
libusbmuxd4:amd64	install
libutempter0:amd64	install
libuuid1:amd64	install
libuv1:amd64	install
libv4l-0:amd64	install
libv4lconvert0:amd64	install
libva-drm2:amd64	install
libva-wayland2:amd64	install
libva-x11-2:amd64	install
libva2:amd64	install
libvariable-magic-perl	install
libvdpa1:amd64	install
libvisio-0.1-1:amd64	install
libvisual-0.4-0:amd64	install
libvncclient1:amd64	install
libvolume-key1	install
libvorbis0a:amd64	install
libvorbisenc2:amd64	install
libvorbisfile3:amd64	install
libvpx5:amd64	install
libvte-2.91-0:amd64	install
libvte-2.91-common	install
libwacom-bin	install
libwacom-common	install

libwacom2:amd64	install
libwavpack1:amd64	install
libwayland-client0:amd64	install
libwayland-cursor0:amd64	install
libwayland-egl1:amd64	install
libwayland-egl1-mesa:amd64	install
libwayland-server0:amd64	install
libwbclient0:amd64	install
libwebkit2gtk-4.0-37:amd64	install
libwebp6:amd64	install
libwebpdemux2:amd64	install
libwebpmux3:amd64	install
libwebrtc-audio-processing1:amd64	install
libwhoopsie-preferences0	install
libwhoopsie0:amd64	install
libwind0-heimdal:amd64	install
libwinpr2-2:amd64	install
libwmf0.2-7:amd64	install
libwmf0.2-7-gtk	install
libwnck-3-0:amd64	install
libwnck-3-common	install
libwoff1:amd64	install
libwpd-0.10-10:amd64	install
libwpg-0.3-3:amd64	install
libwps-0.4-4:amd64	install
libwrap0:amd64	install
libwww-perl	install
libwww-robotrules-perl	install
libwxbase3.0-0v5:amd64	install
libwxgtk3.0-gtk3-0v5:amd64	install
libx11-6:amd64	install
libx11-data	install
libx11-protocol-perl	install
libx11-xcb1:amd64	install
libx264-152:amd64	install
libx265-146:amd64	install
libx32asan4	install
libx32atomic1	install
libx32cilkrt5	install
libx32gcc-7-dev	install
libx32gcc1	install
libx32gomp1	install
libx32itm1	install

libx32quadmath0	install
libx32stdc++6	install
libx32ubsan0	install
libxapian30:amd64	install
libxatracker2:amd64	install
libxau6:amd64	install
libxaw7:amd64	install
libxcb-dri2-0:amd64	install
libxcb-dri3-0:amd64	install
libxcb-glx0:amd64	install
libxcb-icccm4:amd64	install
libxcb-image0:amd64	install
libxcb-keysyms1:amd64	install
libxcb-present0:amd64	install
libxcb-randr0:amd64	install
libxcb-render-util0:amd64	install
libxcb-render0:amd64	install
libxcb-res0:amd64	install
libxcb-shape0:amd64	install
libxcb-shm0:amd64	install
libxcb-sync1:amd64	install
libxcb-util1:amd64	install
libxcb-xfixes0:amd64	install
libxcb-xkb1:amd64	install
libxcb-xv0:amd64	install
libxcb1:amd64	install
libxcomposite1:amd64	install
libxcursor1:amd64	install
libxdamage1:amd64	install
libxdmcp6:amd64	install
libxext6:amd64	install
libxfixes3:amd64	install
libxfont2:amd64	install
libxft2:amd64	install
libxi6:amd64	install
libxinerama1:amd64	install
libxkbcommon-x11-0:amd64	install
libxkbcommon0:amd64	install
libxkbfile1:amd64	install
libxklavier16:amd64	install
libxml-libxml-perl	install
libxml-namespacesupport-perl	install
libxml-parser-perl	install

libxml-sax-base-perl	install
libxml-sax-expat-perl	install
libxml-sax-perl	install
libxml-simple-perl	install
libxml-twig-perl	install
libxml-xpathengine-perl	install
libxml2:amd64	install
libxmlsec1:amd64	install
libxmlsec1-nss:amd64	install
libxmu6:amd64	install
libxmuu1:amd64	install
libxpm4:amd64	install
libxrandr2:amd64	install
libxrender1:amd64	install
libxres1:amd64	install
libxshmfence1:amd64	install
libxslt1.1:amd64	install
libxss1:amd64	install
libxt6:amd64	install
libxtables12:amd64	install
libxtst6:amd64	install
libxv1:amd64	install
libxvidcore4:amd64	install
libxvmc1:amd64	install
libxxf86dga1:amd64	install
libxxf86vm1:amd64	install
libyajl2:amd64	install
libyaml-0-2:amd64	install
libyaml-libyaml-perl	install
libyelp0:amd64	install
libzeitgeist-2.0-0:amd64	install
libzmq5:amd64	install
libzstd1:amd64	install
libzvbi-common	install
libzvbi0:amd64	install
licensecheck	install
light-themes	install
lintian	install
linux-base	install
linux-firmware	install
linux-headers-5.4.0-050400	install
linux-headers-5.4.0-050400-generic	install
linux-headers-5.4.13-050413	install

linux-headers-5.4.13-050413-generic	install
linux-image-unsigned-5.4.0-050400-generic	install
linux-image-unsigned-5.4.13-050413-generic	install
linux-libc-dev:amd64	install
linux-modules-5.4.0-050400-generic	install
linux-modules-5.4.13-050413-generic	install
linux-sound-base	install
lm-sensors	install
locales	install
login	install
logrotate	install
lp-solve	install
lsb-base	install
lsb-release	install
lshw	install
lsof	install
ltrace	install
lvm2	install
m4	install
make	install
man-db	install
manpages	install
manpages-dev	install
mawk	install
media-player-info	install
memtest86+	install
mercurial	install
mercurial-common	install
mesa-va-drivers:amd64	install
mesa-vdpau-drivers:amd64	install
mime-support	install
mlocate	install
mobile-broadband-provider-info	install
modemmanager	install
mokutil	install
mount	install
mouse tweaks	install
mpi-default-bin	install
mpi-default-dev	install
mscompress	install
mtools	install
mtr-tiny	install
multiarch-support	install

mutter	install
mutter-common	install
mythes-en-us	install
nano	install
nautilus	install
nautilus-data	install
nautilus-extension-gnome-terminal	install
nautilus-sendto	install
nautilus-share	install
ncdu	install
ncurses-base	install
ncurses-bin	install
ncurses-term	install
net-tools	install
netbase	install
netcat-openbsd	install
netpbm	install
netplan.io	install
network-manager	install
network-manager-config-connectivity-ubuntu	install
network-manager-gnome	install
network-manager-pptp	install
network-manager-pptp-gnome	install
networkd-dispatcher	install
ninja-build	install
notification-daemon	install
nplan	install
ntfs-3g	install
ntp	install
ntpdate	install
nvme-cli	install
ocl-icd-libopencl1:amd64	install
openmpi-bin	install
openmpi-common	install
openprinting-ppds	install
openssh-client	install
openssh-server	install
openssh-sftp-server	install
openssl	install
orca	install
os-prober	install
p11-kit	install
p11-kit-modules:amd64	install

packagekit	install
packagekit-tools	install
parted	install
passwd	install
pastebinit	install
patch	install
patchutils	install
pciutils	install
pcmciautils	install
perl	install
perl-base	install
perl-modules-5.26	install
perl-openssl-defaults:amd64	install
pinentry-curses	install
pinentry-gnome3	install
pkg-config	install
plymouth	install
plymouth-label	install
plymouth-theme-ubuntu-logo	install
plymouth-theme-ubuntu-text	install
policykit-1	install
policykit-desktop-privileges	install
poppler-data	install
poppler-utils	install
popularity-contest	install
powermgmt-base	install
ppa-purge	install
ppp	install
pppconfig	install
pppoeconf	install
pptp-linux	install
printer-driver-brlaser	install
printer-driver-c2esp	install
printer-driver-foo2zjs	install
printer-driver-foo2zjs-common	install
printer-driver-gutenprint	install
printer-driver-hpcups	install
printer-driver-m2300w	install
printer-driver-min12xxw	install
printer-driver-pnm2ppa	install
printer-driver-postscript-hp	install
printer-driver-ptouch	install
printer-driver-pxljr	install

printer-driver-sag-gdi	install
printer-driver-splix	install
procps	install
psmisc	install
publicsuffix	install
pulseaudio	install
pulseaudio-module-bluetooth	install
pulseaudio-utils	install
python	install
python-all	install
python-all-dev	install
python-apt-common	install
python-asn1crypto	install
python-cffi-backend	install
python-crypto	install
python-cryptography	install
python-dbus	install
python-dev	install
python-enum34	install
python-gi	install
python-idna	install
python-ipaddress	install
python-keyring	install
python-keyrings.alt	install
python-ldb:amd64	install
python-matplotlib-data	install
python-minimal	install
python-numpy	install
python-pip	install
python-pip-whl	install
python-pkg-resources	install
python-psutil	install
python-samba	install
python-secretstorage	install
python-setuptools	install
python-six	install
python-talloc	install
python-tdb	install
python-wheel	install
python-wxgtk3.0	install
python-wxversion	install
python-xdg	install
python2.7	install

python2.7-dev	install
python2.7-minimal	install
python3	install
python3-apport	install
python3-apt	install
python3-aptdaemon	install
python3-aptdaemon.gtk3widgets	install
python3-asn1crypto	install
python3-bottle	install
python3-brlapi	install
python3-cairo:amd64	install
python3-certifi	install
python3-cffi-backend	install
python3-chardet	install
python3-commandnotfound	install
python3-crypto	install
python3-cryptography	install
python3-cups	install
python3-cupshelpers	install
python3-cycler	install
python3-dateutil	install
python3-dbus	install
python3-debconf	install
python3-debian	install
python3-defer	install
python3-dev	install
python3-distro-info	install
python3-distupgrade	install
python3-distutils	install
python3-docker	install
python3-dockerpycreds	install
python3-gdbm:amd64	install
python3-gi	install
python3-gi-cairo	install
python3-gpg	install
python3-httplib2	install
python3-idna	install
python3-influxdb	install
python3-keyring	install
python3-keyrings.alt	install
python3-launchpadlib	install
python3-lazr.restfulclient	install
python3-lazr.uri	install

python3-lib2to3	install
python3-louis	install
python3-macaroonbakery	install
python3-magic	install
python3-mako	install
python3-markupsafe	install
python3-matplotlib	install
python3-minimal	install
python3-nacl	install
python3-netifaces	install
python3-newt:amd64	install
python3-numpy	install
python3-oauth	install
python3-olefile	install
python3-pexpect	install
python3-pil:amd64	install
python3-pip	install
python3-pkg-resources	install
python3-ply	install
python3-problem-report	install
python3-protobuf	install
python3-psutil	install
python3-ptyprocess	install
python3-pyasn1	install
python3-pyatspi	install
python3-pycryptodome	install
python3-pyinotify	install
python3-pymacaroons	install
python3-pyparsing	install
python3-pysmi	install
python3-pysnmp4	install
python3-pystache	install
python3-renderrpm:amd64	install
python3-reportlab	install
python3-reportlab-accel:amd64	install
python3-requests	install
python3-requests-unixsocket	install
python3-rfc3339	install
python3-secretstorage	install
python3-setuptools	install
python3-simplejson	install
python3-six	install
python3-software-properties	install

python3-speechd	install
python3-systemd	install
python3-tk:amd64	install
python3-tz	install
python3-unidiff	install
python3-uno	install
python3-update-manager	install
python3-urllib3	install
python3-wadllib	install
python3-websocket	install
python3-wheel	install
python3-xdg	install
python3-xkit	install
python3-yaml	install
python3-zope.interface	install
python3.6	install
python3.6-dev	install
python3.6-minimal	install
qpdf	install
rarian-compat	install
readline-common	install
remmina	install
remmina-common	install
remmina-plugin-rdp:amd64	install
remmina-plugin-secret:amd64	install
remmina-plugin-vnc:amd64	install
rfkill	install
rhythmbox	install
rhythmbox-data	install
rhythmbox-plugin-alternative-toolbar	install
rhythmbox-plugins	install
rsync	install
rsyslog	install
rtkit	install
run-one	install
samba-common	install
samba-common-bin	install
samba-libs:amd64	install
sane-utils	install
sbsigntool	install
seahorse	install
secureboot-db	install
sed	install

sensible-utils	install
session-migration	install
sgml-base	install
sgml-data	install
shared-mime-info	install
shim	install
shim-signed	install
shotwell	install
shotwell-common	install
simple-scan	install
snapd	install
sntp	install
software-properties-common	install
software-properties-gtk	install
sound-icons	install
sound-theme-freedesktop	install
speech-dispatcher	install
speech-dispatcher-audio-plugins:amd64	install
speech-dispatcher-espeak-ng	install
spice-vdagent	install
squashfs-tools	install
ssh	install
ssh-import-id	install
ssl-cert	install
strace	install
sudo	install
synaptic	install
syslinux	install
syslinux-common	install
syslinux-legacy	install
system-config-printer	install
system-config-printer-common	install
system-config-printer-udev	install
systemd	install
systemd-sysv	install
sysvinit-utils	install
tlutils	install
tar	install
tcpdump	install
telnet	install
thunderbird	install
thunderbird-gnome-support	install
thunderbird-locale-en	install

thunderbird-locale-en-us	install
time	install
tk8.6-blt2.5	install
tmux	install
totem	install
totem-common	install
totem-plugins	install
transmission-common	install
transmission-gtk	install
ttf-bitstream-vera	install
tzdata	install
ubuntu-adantage-tools	install
ubuntu-artwork	install
ubuntu-desktop	install
ubuntu-docs	install
ubuntu-drivers-common	install
ubuntu-keyring	install
ubuntu-minimal	install
ubuntu-mono	install
ubuntu-release-upgrader-core	install
ubuntu-release-upgrader-gtk	install
ubuntu-report	install
ubuntu-restricted-addons	install
ubuntu-session	install
ubuntu-settings	install
ubuntu-software	install
ubuntu-sounds	install
ubuntu-standard	install
ubuntu-system-service	install
ubuntu-wallpapers	install
ubuntu-wallpapers-bionic	install
ubuntu-web-launchers	install
ucf	install
udev	install
udisks2	install
ufw	install
unattended-upgrades	install
uno-libs3	install
unzip	install
update-inetd	install
update-manager	install
update-manager-core	install
update-notifier	install

update-notifier-common	install
upower	install
ure	install
ureadahead	install
usb-creator-common	install
usb-creator-gtk	install
usb-modeswitch	install
usb-modeswitch-data	install
usbmuxd	install
usbutils	install
util-linux	install
uuid-runtime	install
va-driver-all:amd64	install
vdpau-driver-all:amd64	install
vim	install
vim-common	install
vim-runtime	install
vim-tiny	install
vino	install
wamerican	install
wbritish	install
wdiff	install
wget	install
whiptail	install
whois	install
whoopsie	install
whoopsie-preferences	install
wireless-regdb	install
wireless-tools	install
wpasupplicant	install
x11-apps	install
x11-common	install
x11-session-utils	install
x11-utils	install
x11-xkb-utils	install
x11-xserver-utils	install
xauth	install
xbitmaps	install
xbrlapi	install
xcursor-themes	install
xdg-desktop-portal	install
xdg-desktop-portal-gtk	install
xdg-user-dirs	install

xdg-user-dirs-gtk	install
xdg-utils	install
xfonts-base	install
xfonts-encodings	install
xfonts-scalable	install
xfonts-utils	install
xinit	install
xinput	install
xbps	install
xml-core	install
xorg	install
xorg-docs-core	install
xserver-common	install
xserver-xephyr	install
xserver-xorg-core-hwe-18.04	install
xserver-xorg-hwe-18.04	install
xserver-xorg-input-all-hwe-18.04	install
xserver-xorg-input-libinput-hwe-18.04	install
xserver-xorg-input-wacom-hwe-18.04	install
xserver-xorg-legacy-hwe-18.04	install
xserver-xorg-video-all-hwe-18.04	install
xserver-xorg-video-amdgpu-hwe-18.04	install
xserver-xorg-video-ati-hwe-18.04	install
xserver-xorg-video-fbdev-hwe-18.04	install
xserver-xorg-video-intel-hwe-18.04	install
xserver-xorg-video-nouveau-hwe-18.04	install
xserver-xorg-video-qxl-hwe-18.04	install
xserver-xorg-video-radeon-hwe-18.04	install
xserver-xorg-video-vesa-hwe-18.04	install
xserver-xorg-video-vmware-hwe-18.04	install
xul-ext-ubufox	install
xwayland	install
xxd	install
xz-utils	install
yelp	install
yelp-xsl	install
zeitgeist-core	install
zenity	install
zenity-common	install
zip	install
zlib1g:amd64	install
zlib1g-dev:amd64	install

Procedure to enable Application Direct (AD) mode

- Once ndctl and ipmctl installed, we can use ipmctl to show the information about nvdimm in the system:

```
$ sudo ipmctl show -dimm
```

```
DimmID Capacity HealthState ActionRequired LockState FWVersion
0x0020 125.7 GiB Healthy 0 Disabled 01.00.00.4351
pmctl can show information about nvdimm resource allocation:
```

```
$ sudo ipmctl show -memoryresourcesCapacity=125.7 GiB
```

```
MemoryCapacity=125.0 GiB
AppDirectCapacity=0.0 GiB
UnconfiguredCapacity=0.0 GiB
InaccessibleCapacity=0.0 GiB
ReservedCapacity=0.7 GiB
```

- Memory mode, and the next step is to enable App Direct with full capacity. Before that, let's make sure if nvdimm has a capability to support AD or not. Likewise, ipmctl is used to show its capabilities:

```
$ sudo ipmctl show -a -system -capabilities
```

```
PlatformConfigSupported=1
Alignment=1.0 GiB
AllowedVolatileMode=Memory Mode
CurrentVolatileMode=Memory Mode
AllowedAppDirectMode=App Direct
ModesSupported=1LM, Memory Mode, App Direct
SupportedAppDirectSettings=x1 (ByOne), x2 - 4KB iMC x 4KB
Channel (4KB_4KB), x3 - 4KB iMC x 4KB Channel (4KB_4KB), x4 -
4KB iMC x 4KB Channel (4KB_4KB), x6 - 4KB iMC x 4KB Channel
(4KB_4KB)
RecommendedAppDirectSettings=x1 (ByOne), x2 - 4KB iMC x 4KB
Channel (4KB_4KB), x3 - 4KB iMC x 4KB Channel (4KB_4KB), x4 -
4KB iMC x 4KB Channel (4KB_4KB), x6 - 4KB iMC x 4KB Channel
(4KB_4KB)
MinNamespaceSize=1.0 GiB
AppDirectMirrorSupported=0
DimmSpareSupported=0
```

```
AppDirectMigrationSupported=0
RenameNamespaceSupported=1
GrowAppDirectNamespaceSupported=0
ShrinkAppDirectNamespaceSupported=0
InitiateScrubSupported=0
AdrSupported=1
EraseDeviceDataSupported=0
EnableDeviceSecuritySupported=0
DisableDeviceSecuritySupported=1
UnlockDeviceSecuritySupported=0
FreezeDeviceSecuritySupported=0
ChangeDevicePassphraseSupported=0
```

- In order to make use nvdimm with AD mode, Intel ipmctl takes a responsibility for management but ndctl also tackles with namespace management. The conceptional workflow of creating a MM/AD mode is as below:
 - Create a memory allocation goal
 - Reboot system
 - Create a namespace for specific regions
- To create a goal with 100% AD, just issue the commands below.
Before system reboots, the goal information can be printed by \$ sudo ipmctl show -a -goal

```
$ sudo ipmctl create -goal

The following configuration will be applied:SocketID
DimmID MemorySize AppDirect1Size AppDirect2Size
0x0000 0x0020 0.0 GiB 125.0 GiB 0.0 GiB
Do you want to continue? [y/n] y
Created following region configuration goal
SocketID DimmID MemorySize AppDirect1Size
AppDirect2Size
0x0000 0x0020 0.0 GiB 125.0 GiB 0.0 GiB
A reboot is required to process new memory allocation
goals.
```

- After reboot, use ipmctl again to confirm if the goal was executed successfully:

```
$ sudo ipmctl show -memoryresourcesCapacity=125.7 GiB

MemoryCapacity=0.0 GiB
AppDirectCapacity=125.0 GiB
UnconfiguredCapacity=0.6 GiB
InaccessibleCapacity=0.0 GiB
ReservedCapacity=0.0 GiB
```

- Assuming the goal worked correctly, and then creating a namespace for goal is as below:

```
$ sudo ndctl create-namespace

{
    "dev": "namespace0.0",
    "mode": "fsdax",
    "map": "dev",
    "size": "123.04 GiB (132.12 GB)",
    "uuid": "c3991a7f-deb1-4d0f-8379-80047dc63821",
    "raw_uuid": "8212fffc1-8f66-4ff7-bb22-198160e71170",
    "sector_size": 512,
    "blockdev": "pmem0",
    "numa_node": 0
}
```

- The namespace function will create a blockdev node (e.g. /dev/pmem0) responding to the specific region. In this case, a namespace is generated with full capacity of nvdimm. Furthermore, using additional parameter “-s” is to specify the requested region per command so that the multiple nodes would be generated simultaneously.
- Now /dev/pmem0 exists in the system, the next step is to make everything as usual. For example, partitioning pmem0 via parted could be

```
$ parted -s -a optimal /dev/pmem0 \
    mklabel gpt -- \
    mkpart primary ext4 1MiB 4GiB \
    mkpart primary xfs 4GiB 12GiB \
```

```
mkpart primary btrfs 12GiB -1MiB \
print
```

- Or, make a specific filesystem on /dev/pmem0 easily:

```
$ sudo mkfs.xfs /dev/pmem0
```

- For mounting file system, ext4 and xfs support DAX. To get the benefit from it, the extra option needs to be added in the mount:

```
$ mkdir xfs-pmem0
$ sudo mount -o dax /dev/pmem0 xfs-pmem0
```

- If /dev/pmem0 is successfully mounted, the dmesg should give the following messages:

```
$ mount | grep dax
/dev/pmem0 on /home/u/xfs-pmem0 type xfs
(rw,relatime,attr2,dax,inode64,noquota)$ dmesg | grep
pmem
[92747.084077] XFS (pmem0): Mounting V5 Filesystem
[92747.087630] XFS (pmem0): Ending clean mount$ dmesg | grep
DAX
[ 1580.768764] XFS (pmem0): DAX enabled. Warning:
EXPERIMENTAL, use at your own risk
```

Mixed mode

The mixed mode is still painless to manipulate ipmctl to give the specific percentage for memory mode (AD is just opposite value). For example, the configuration for MM:AD is set to 50:50, and the command will be issued by

```
$ ipmctl create -goal memorymode=50
persistentmemorytype=appdirect
$ ipmctl create -goal memorymode=50
persistentmemorytype=appdirectnotinterleaved
```

In order to allow applications access persistent memory as memory-mapped files, the PMDK [18]

References

1. https://ark.intel.com/content/www/us/en/ark/search/featurefilter.html?productType=873&2_OptaneDCPersistentMemoryVersion=True
2. Intel Optane DC 256GB Persistent Memory Model
NMA1XXD256GPSU4.
https://mysamples.intel.com/SAM_U_Product/ProductSearch.aspx?SearchText=APACHE%20PASS
3. <https://software.intel.com/en-us/articles/quick-start-guide-configure-intel-optane-dc-persistent-memory-on-linux>
4. <https://docs.pmem.io/getting-started-guide/what-is-ndctl>
5. <https://pmem.io/pmdk/>
6. How to: Get Apache Pass AEP NVDimm to work on Purley Refresh.
<https://nsg-wiki.intel.com/display/NSM/How+to%3A+Get+Apache+Pass+AEP+NVDimm+to+work+on+Purley+Refresh>
7. Optane DIMMs Memory BKCs
<https://www.intel.com/content/dam/support/us/en/documents/memory-and-storage/data-center-persistent-mem/Population-Configuration.pdf>
8. Intel CPUs supporting Optane DIMMs
https://ark.intel.com/content/www/us/en/ark/search/featurefilter.html?productType=873&2_OptaneDCPersistentMemoryVersion=True
9. NSF-Intel CAPA Information
10. Operating Systems Supported
<https://www.intel.sg/content/www/xa/en/support/articles/000032860/memory-and-storage/data-center-persistent-memory.html?countrylabel=Asia%20Pacific>
11. Ubuntu Setup Guidance. <https://medium.com/@woodrow.shen/enable-intel-nvdimmon-ubuntu-6498168f0cb1>
12. <https://nvdimm.wiki.kernel.org>
13. <https://www.kernel.org/doc/Documentation/nvdimm/btt.txt>
14. <https://www.kernel.org/doc/Documentation/filesystems/dax.txt>
15. <https://git.kernel.org/pub/scm/linux/kernel/git/nvdimm/nvdimm.git>
16. <https://github.com/pmem/ndctl>
17. <https://github.com/intel/ipmctl>
18. <http://pmem.io>

19. ZFS Setup. <https://ubuntu.com/tutorials/setup-zfs-storage-pool#1-overview>
20. VMware performance.
<https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/techpaper/performance/bigdata-vsphere65-perf.pdf>
21. Ubuntu Performance
https://wiki.mikejung.biz/Ubuntu_Performance_Tuning
22. <https://www.monitis.com/blog/20-linux-server-performance-tips-part1/>
23. Redhat performance tuning guide.
https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/pdf/performance_tuning_guide/Red_Hat_Enterprise_Linux-7-Performance_Tuning_Guide-en-US.pdf
24. Linux Performance and Tuning Guidelines by Eduardo Ciliendo Takechika Kunimasa. Lenovo Performance tuning guide.
<https://lenovopress.com/redp4285.pdf>
25. <https://www.linuxtechi.com/set-ulimit-file-descriptors-limit-linux-servers/>
26. IBM ulimit recommendations
27. https://www.ibm.com/support/knowledgecenter/SSEP7J_11.1.0/com.ibm.swg.ba.cognos.inst_cr_winux.doc/c_inst_ulimitsettingsonunixandlinuxoperatingsystems.html
28. https://www.ibm.com/support/knowledgecenter/SS8NLW_10.0.0/com.ibm.discovery.es.in.doc/iiysiulimits.htm
29. Intel Server Board S2600WF Product Family Technical Product Specification
https://www.intel.com/content/dam/support/us/en/documents/server-products/server-boards/S2600WF_TPS.pdf
30. Intel Server Board S2600WF Family BIOS and Firmware Update Package for UEFI
<https://downloadcenter.intel.com/download/29105/Intel-Server-Board-S2600WF-Family-BIOS-and-Firmware-Update-Package-for-UEFI?product=77593>
31. Include System BIOS, ME Firmware, BMC Firmware, FRU SDR, and Intel Optane Persistent memory Firmware
32. Performance configuration options
<https://software.intel.com/content/www/us/en/develop/articles/speedin>

[g-up-io-workloads-with-intel-optane-dc-persistent-memory-modules.html](#)

33. Steve Scargall. Programming Persistent Memory: A Comprehensive Guide for Developers. <https://www.amazon.com/Programming-Persistent-Memory-Comprehensive-Developers/dp/1484249313>
34. Microsoft Persistent Memory Server. <https://docs.microsoft.com/en-us/windows-server/storage/storage-spaces/deploy-pmem>
35. Redhat. https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/storage_administration_guide/ch-persistent-memory-nvdimms
36. Intel Install Guide. <https://docs.pmem.io/persistent-memory/getting-started-guide/installing-ndctl>
37. Training Video.
<https://software.intel.com/content/www/us/en/develop/videos/provisioning-intel-optane-dc-persistent-memory-modules-in-linux.html>
38. VROC.
<https://www.intel.com/content/dam/support/us/en/documents/memory-and-storage/ssd-software/VROC-Ubuntu-Setup-UserGuide-342787-US.pdf>
39. <https://www.supermicro.com/en/products/motherboard/X11QPH+>

Documentation Sphinx Generator for Linux

A simple generator for using Sphinx to document small Python modules.

reStructured Text cheat sheet

- <http://github.com/#index>

Python documentation cheat sheet

module/__init__.py

Installation

```
$ sudo apt-get install python-sphinx
$ sudo pip install sphinx
# Depends on which version you prefer ...
$ sudo pip3 install sphinx
```

Quickstart

Sphinx offers an easy quickstart

```
$ mkdir docs
$ cd docs
# Quickstart, select yes for apidoc and mathjax and for
splitting build and source.
$ sphinx-quickstart
$ sphinx
```

Choose to separate source and build directories, choose project name and version and the autodoc extension.

If the code/module to be documentation is accessible from the root directory, edit

```
docs/source/conf.py
```

as follows

```
import os
import sys
sys.path.insert(0, os.path.abspath('..../'))
```

Then the modules can be automatically documented using:

```
$ sphinx-apidoc -f -o source/ ../
$ make html
```

Python 3.8

For modules or dependencies not supporting Python 3, *docs/Makefile* can be adapted:

```
SPHINXBUILD = python -c "import
sys,sphinx;sys.exit(sphinx.main(sys.argv))"
```

Instruction Commands

New development should have a unit test capability built in to ensure there are no regressions.

- Auto doxygen location
 - RAAD/dox/build/index.html
 - RAAD/dox/build/epub/RAAD.epub
- Docstring Style
 - https://sphinxcontrib-napoleon.readthedocs.io/en/latest/example_google.html
- Example documentation execute order:

```
cd RAAD/dox/source/
python findClasses.py
cd ..
```

```
make clean  
make
```

Operating System, Virtual Machine, Container Setup Options

Choices for setups are:

1. **Native OS Ubuntu 22.04 LTS** [#_Option_1]
2. **Ubuntu 22.04 LTS Docker** [#_Option_2]
3. **Create your own image of Ubuntu 22.04 LTS** [#_Option_3]
4. **Use a partial make image of Ubuntu 22.04 LTS** [#_Option_4]
5. **Intel Pre-made image of Ubuntu 22.04 LTS** [#_Option_5]
6. **Windows 10 x86_64 (Not Recommended)** [#_Option_6]

Option 1 Ubuntu 22.04 LTS

1. See [#_Clone_Repository-label]
2. See [#_Install_RAAD_Ubuntu_Script_for_Requirements-label]
3. See [#_Use_default_IDE_to_run_RAAD_main-label]

Option 2 Ubuntu 22.04 LTS + Docker

Option 3 Create your own image of Ubuntu 22.04 LTS

1. Create Ubuntu download image at:
 - o <https://ubuntu-mate.org/download/amd64/jammy/>
2. To use the virtual machine download VMWare workstation player
 - o <https://www.vmware.com/products/workstation-player/workstation-player-evaluation.html>
3. Follow steps:
 - o <https://kb.vmware.com/s/article/1018415>

- <https://kb.vmware.com/s/article/1018414>
- 4. See [#_Clone_Repository-label]
- 5. See [#_Install_RAAD_Ubuntu_Script_for_Requirements-label]
- 6. See [#_Use_default_IDE_to_run_RAAD_main-label]

Option 4 Use a partial make image of Ubuntu 22.04 LTS

1. To download semi-pre-made image:
 - <https://sourceforge.net/projects/osboxes/files/v/vm/55-U--u/22.04/64bit.7z/download>
2. Login info is located at:
 - <https://www.osboxes.org/faq/what-are-the-credentials-for-virtual-machine-image/>
3. Install virtual machine tools
 - <https://kb.vmware.com/s/article/1018414>
4. Proceed to 'Clone Repository'
5. See [#_Clone_Repository-label]
6. See [#_Install_RAAD_Ubuntu_Script_for_Requirements-label]
7. See [#_Use_default_IDE_to_run_RAAD_main-label]

Option 5 Intel Pre-made image of Ubuntu 22.04 LTS

1. Download VMWare workstation player
 - <https://www.vmware.com/products/workstation-player/workstation-player-evaluation.html>
2. Decompress the image
 - To decompress the virtual machine download and install 7-Zip
 - <https://www.7-zip.org/download.html>
 - Decompress the image/RAAD/vm_ubuntu_22.04_x86_64/vm.7z

3. Run RAAD

- Once installed open vmware workstation player and select File -> Open...

4. Select the file:

RAAD/vm_ubuntu_22.04_x86_64/RAAD_Testing_U22LTSx8664/RAAD_Testing_U22LTSx8664.vmx

- User and password pairs are:
- raad_admin, raad
- raaduser, raad
- developer, raad_dev
- tester, raad_tester

5. Login to raaduser

- Note Anaconda is installed at: /opt/anaconda3
- Symbolic link is: /home/raad/anaconda3 ->/opt/anaconda3

6. See [#_Clone_Repository-label]

7. See [#_Install_RAAD_Ubuntu_Script_for_Requirements-label]

8. See [#_Use_default_IDE_to_run_RAAD_main-label]

Clone Repository

Right click on desktop and select 'Open in Terminal'

```
cd ~/Desktop  
mkdir -p github  
cd github  
git clone --recursive https://github.com/intel/RAAD.git
```

Install RAAD Ubuntu Script for Requirements

1. Install script is at

https://github.com/intel/RAAD/blob/main/vm_ubuntu_22.04_x86_64/UbuntuInstall.sh

2. To install requirements run the script open a terminal and type

```
cd RAAD
chmod +x UbuntuInstall.sh
./UbuntuInstall.sh
```

Use default IDE to run RAAD main.py

- A setup IDE is PyCharm community
 - Menu -> Programming -> PyCharm Community Edition
 - Within PyCharm select File-> Open
 - I.E. '/home/raaduser/Desktop/github/RAAD/'
 - To run the entry point of RAAD is:
 - I.E.
'/home/raaduser/Desktop/github/RAAD/src/main.py'
 - Right click on main.py then Debug 'main.py'
 - The default parameters will open the GUI interface.

Option 6 Windows 10 x86_64 (Not Recommended)

Windows 10 x64

Installation

- Open command prompt

```
set url=https://repo.anaconda.com/archive/Anaconda3-
2020.07-Windows-x86_64.exe
set file=Anaconda3-2020.07-Windows-x86_64.exe
certutil -urlcache -split -f %url% %file%
start /wait "" Anaconda3-2020.07-Windows-x86_64.exe
/InstallationType=JustMe /RegisterPython=0 /S
/D=%UserProfile%\anaconda3
start /wait ""
%UserProfile%\anaconda3\Scripts\activate.bat
```

- Anaconda should be open or open Anaconda manually
- Change to repository directory

- Recreate and update the base environment with the best known method.
 - Please note `environment_windows-x86_64.yml` should be `environment_{operating system}.yml`

```
conda update --force conda -y
conda update anaconda -y
conda update --all
conda update anaconda-navigator
conda update python
conda env create --file environment_win-x86_64.yml
```

- To update use:

```
conda env update -f environment_win-x86_64.yml
```

Pycharm Debugger

Client Config via Command Line

- Create file: `silent.config` using create config file via command line

```
echo. > "silent.config"
echo mode=user >> "silent.config"
echo launcher32=0 >> "silent.config"
echo launcher64=1 >> "silent.config"
echo updatePATH=0 >> "silent.config"
echo jre32=1 >> "silent.config"
echo updateContextMenu=1 >> "silent.config"
echo python2=0 >> "silent.config"
echo python3=0 >> "silent.config"
echo regenerationSharedArchive=1 >> "silent.config"
echo .py=0 >> "silent.config"
```

- Optionally, edit config file manually:

```
; Installation mode. It can be user or admin.
; NOTE: for admin mode please use "Run as
Administrator" for command prompt to avoid UAC dialog
or user 'admin'.
mode=user
```

```

; Desktop shortcut for launchers
launcher32=0
launcher64=1

; Add launchers path to PATH env variable
updatePATH=0

; Download and install jre32. This may take a few
minutes.
jre32=1

; Add "Open Folder as Project" to context menu
updateContextMenu=1

; Download and install python. This may take a few
minutes.
python2=0
python3=0

; Regenerating the Shared Archive
; https://docs.oracle.com/en/java/javase/11/vm/class-
data-sharing.html
regenerationSharedArchive=1

; List of associations. To create an association change
value to 1.
.py=0

```

Commandline Install

- Installing in command line

```

set url=https://download.jetbrains.com/python/pycharm-
community-2021.2.2.exe
set file=pycharm-community-2021.2.2.exe
certutil -urlcache -split -f %url% %file%
start /wait "" pycharm-community-2021.2.2.exe /S
/CONFIG=.\silent.config
/LOG=C:\JetBrains\PyCharmEdu\install.log
/D=C:\JetBrains\Edu\PyCharm_2020

```

RAAD Executable Installer for Windows 10 x86_64

If it is your first time utilizing RAAD, please follow the instructions below to download and execute the required files to start developing or utilizing RAAD. There are two sections in this article:

- one for developers.
- one for users of RAAD.

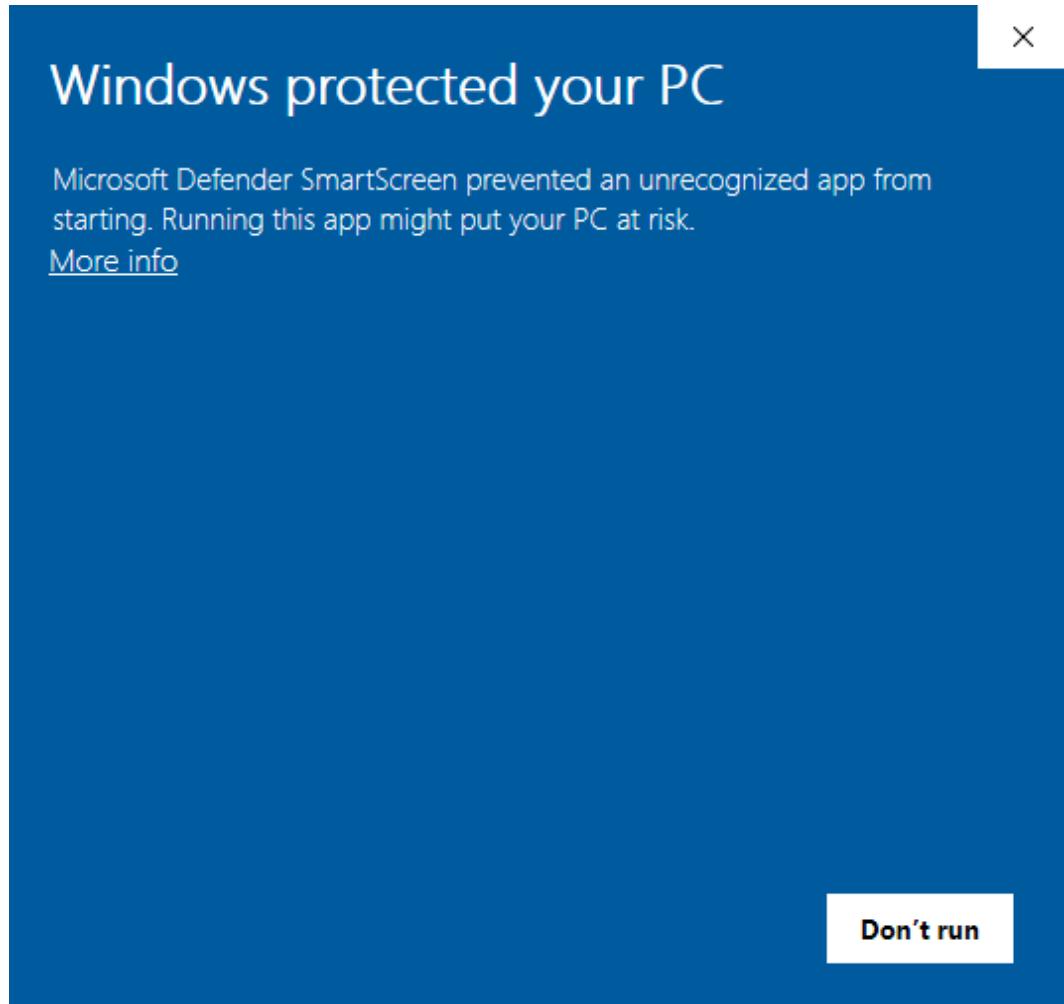
The section for developers is directly below. The section for users can be found after the section for developers. Please note the tutorial is for Windows, the tutorial for Ubuntu/Linux will be uploaded at a later iteration of the project.

Information for Developers of RAAD:

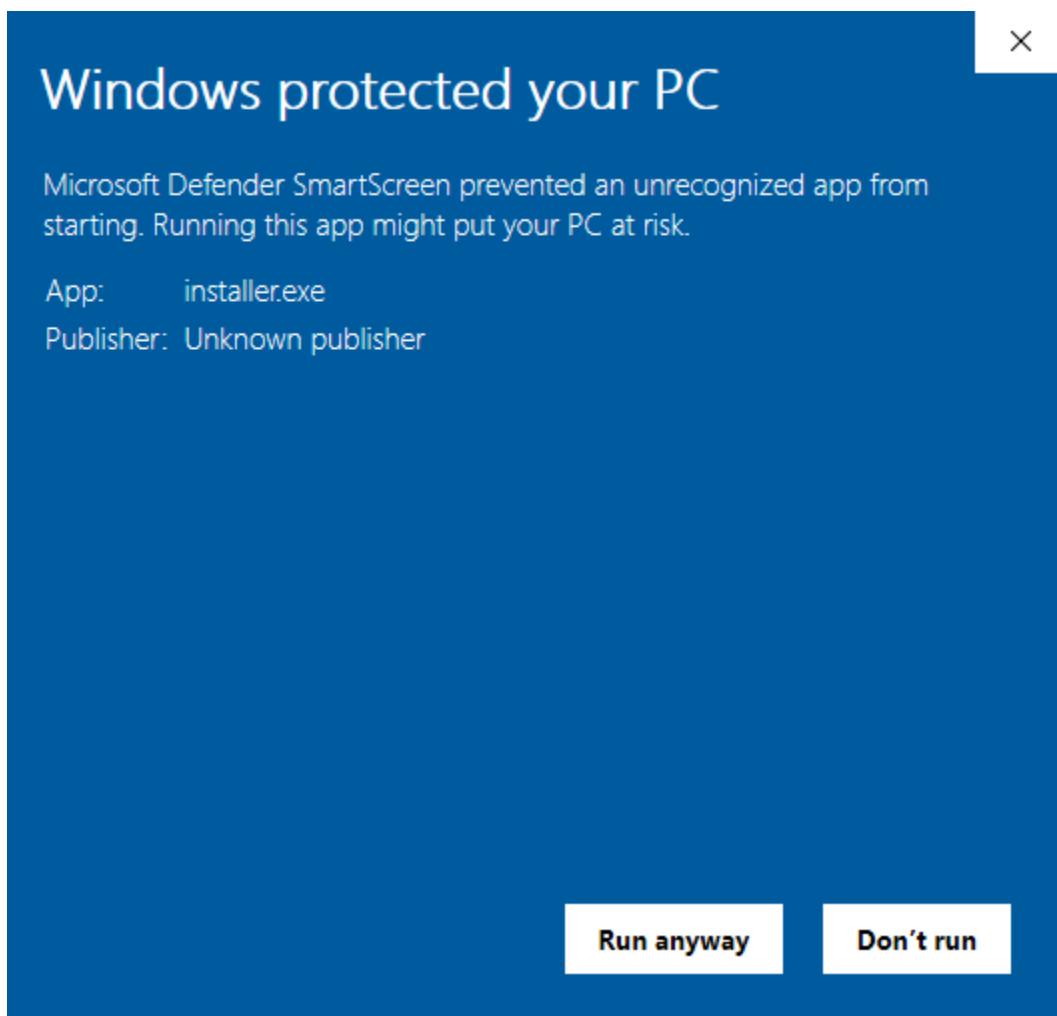
- As a developer you need to download the installation wizard found in the RAAD under "RAAD Executables": The wizard is named installer.exe. Once you have downloaded the file from teams, please follow the instructions below. @todo jdtarang

Instructions for the installation wizard:

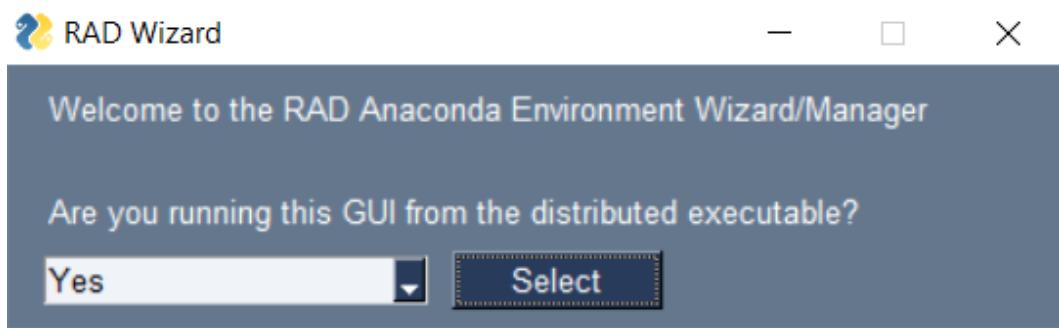
After the download is done, run the executable. You may encounter a blue window telling you "Windows protected your PC".



Click on "More info" and then on the new button "Run Anyway".

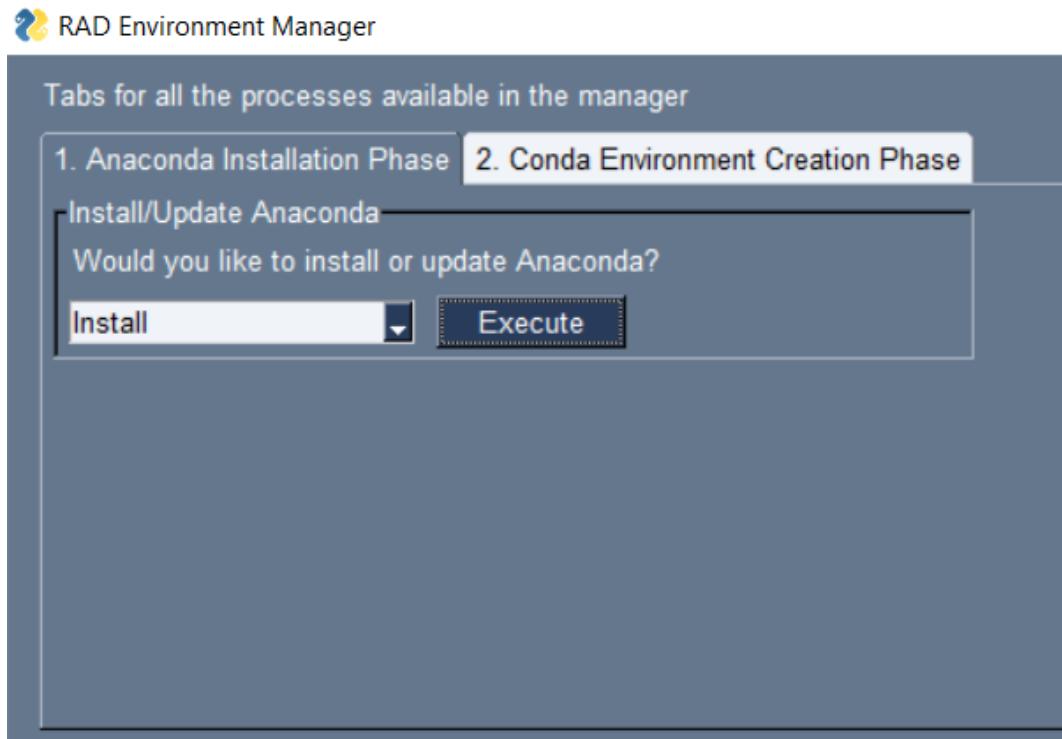


Once the executable is running, you should see a background terminal window with all the warning and info statements for the wizard executable. If the executable runs correctly, a window like the one shown below should pop-up on top of the terminal window. Select "Yes" on the first drop down menu (as the wizard is being run directly from the executable), and continue with the other steps.

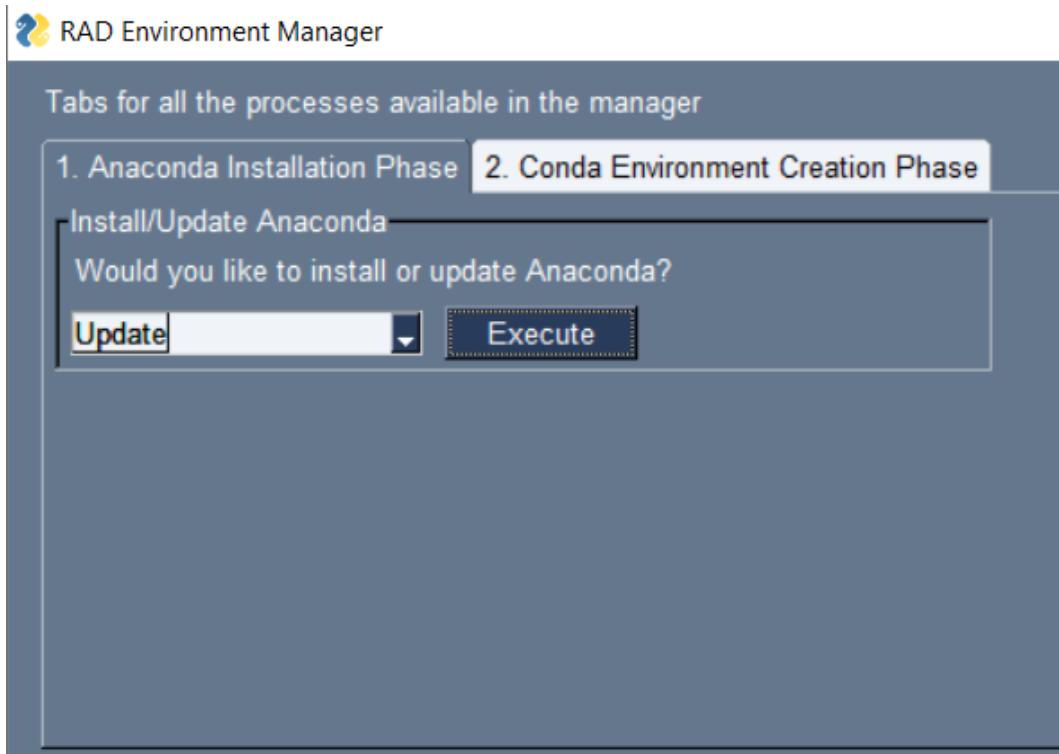


Note: Make sure you click on "Select" so that the GUI registers the option chosen in the drop down menu.

After you have accessed the second screen, install Anaconda by choosing the "install" option from the drop down menu and clicking the "Execute" button shown below. The Anaconda executable will be downloaded and the installer will be launched. Accept the terms and conditions and don't change any of the options while installing anaconda (click on next until you reach the installation). Wait for the installation to finish and then return to RAAD installation Wizard



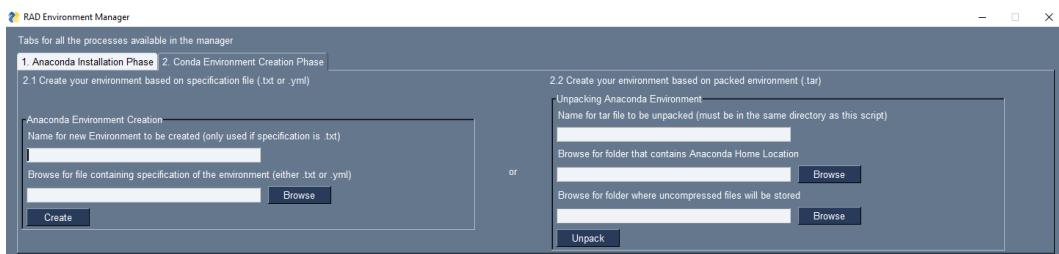
Note: If for some reason you already had Anaconda installed in your device, you can update it to the latest version by choosing the "update" option in the drop down menu and clicking the "Execute" button shown below.



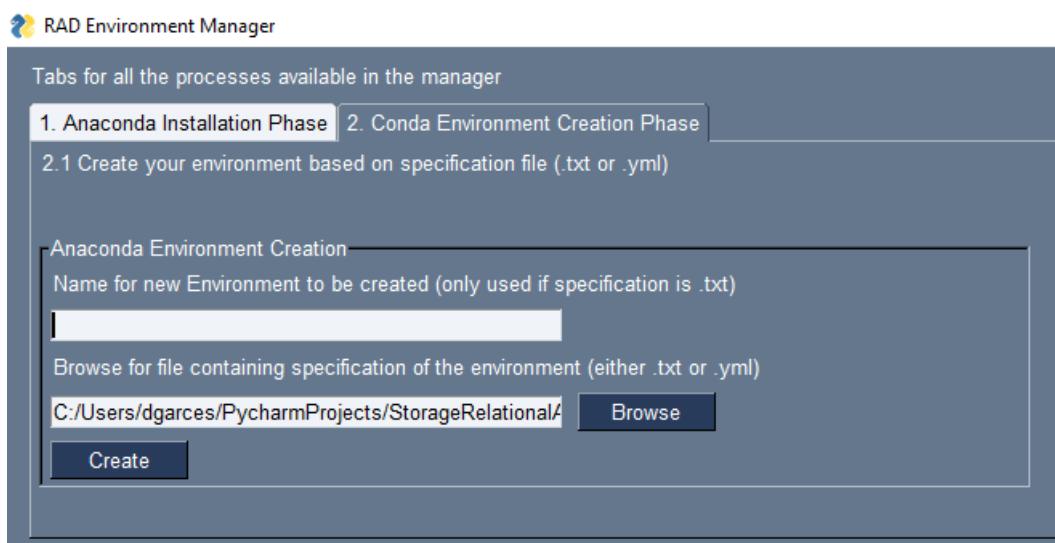
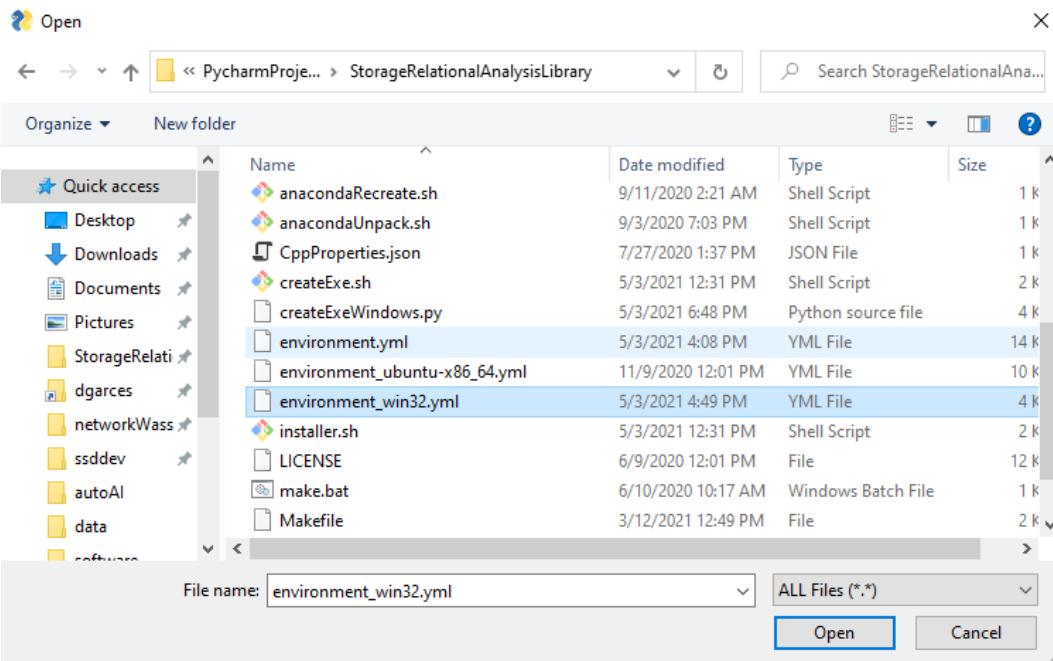
Note: If the wizard starts crashing before Anaconda gets updated/installed, please check your connection to the internet and try again. If the problem persists, please manually download anaconda by going to
<https://www.anaconda.com/products/individual>

Download the environment configuration file found here:
`environment_win-x86_64.yml`

After the file has been download into your local machine, click on the tab titled "2. Conda Environment Creation Phase" to change to the next tab and start the environment creation process. The tab looks like the one displayed below



Once you are in the "Conda Environment Creation Phase" tab, use the "Anaconda Environment Creation" tool to create the environment using the .yml that you downloaded in step 3. To create the environment, you need to click on the "Browse" button, which will display a navigator window as shown below. Use this navigation window to locate your copy of "environment_win32.yml" and click the open button to insert the path in the input line.



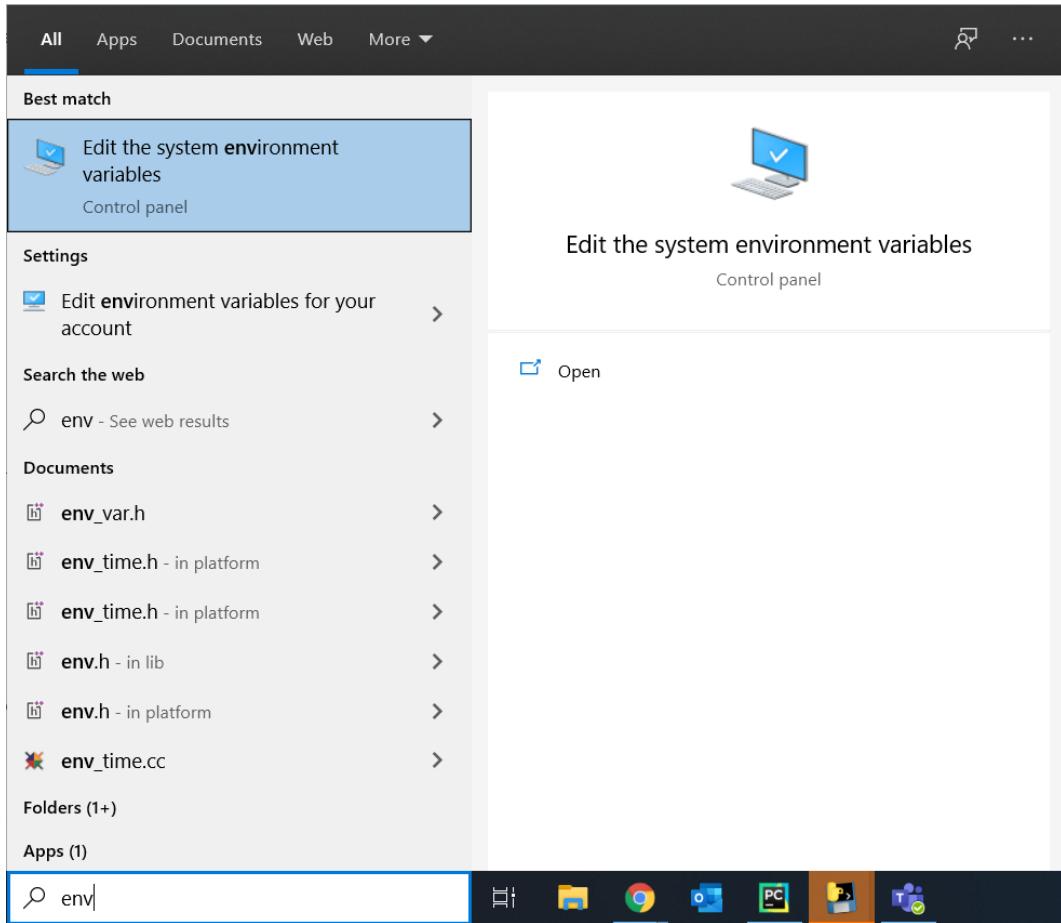
Once the path is in the input line, click on the create button and wait for the process to finish. If the environment was created successfully, a success message like the one displayed below will be shown in the background terminal window.

```
Successfully installed absl-py-0.11.0 alabaster-0.7.12 altgraph-0.17 asn1crypto-1.4.0 astunparse-1.6.3 babe-1-2.9.0 cachetools-4.1.1 cffi-1.14.5 chardet-3.0.4 colorama-0.4.4 cryptography-3.4.7 docutils-0.16 flatbuffers-1.12 future-0.18.2 gast-0.3.3 gnupg-2.3.1 google-auth-1.23.0 google-auth-oauthlib-0.4.2 google-pasta-0.2.0 grpcio-1.32.0 h5py-2.10.0 idna-2.10 imagesize-1.2.0 importlib-metadata-2.0.0 jinja2-2.11.3 keras-preprocessing-1.1.2 markdown-3.3.3 markupsafe-1.1.1 naked-0.1.31 numpy-1.19.5 oauthlib-3.1.0 opt-einsum-3.3.0 ordered-set-4.0.2 packaging-20.9 pfile-2019.4.18 protobuf-3.13.0 pyasn1-0.4.8 pyasn1-modules-0.2.8 pycparser-2.20 pycryptodome-3.10.1 pygments-2.8.0 pyinstaller-hooks-contrib-2021.1 pylatex-1.4.1 pwini32-3000 pywin32-ctypes-0.2.0 pyyaml-5.3.1 requests-2.25.0 requests-oauthlib-1.3.0 rsa-4.6 shellescape-3.8.1 snowballstemmer-2.1.0 sphinx-3.5.1 sphinxcontrib-applehelp-1.0.2 sphinxcontrib-devhelp-1.0.2 sphinxcontrib-htmlhelp-1.0.3 sphinxcontrib-jsmath-1.0.1 sphinxcontrib-qthelp-1.0.3 sphinxcontrib-serializinghtml-1.1.4 tensorboard-2.4.0 tensorflow-plugin-wit-1.7.0 tensorflow-2.4.1 tensorflow-estimator-2.4.0 termcolor-1.1.0 tinaaes-1.0.1 typing-extensions-3.7.4.3 urllib3-1.26.1 werkzeug-1.0.1 wrapt-1.12.1 zipp-3.4.0

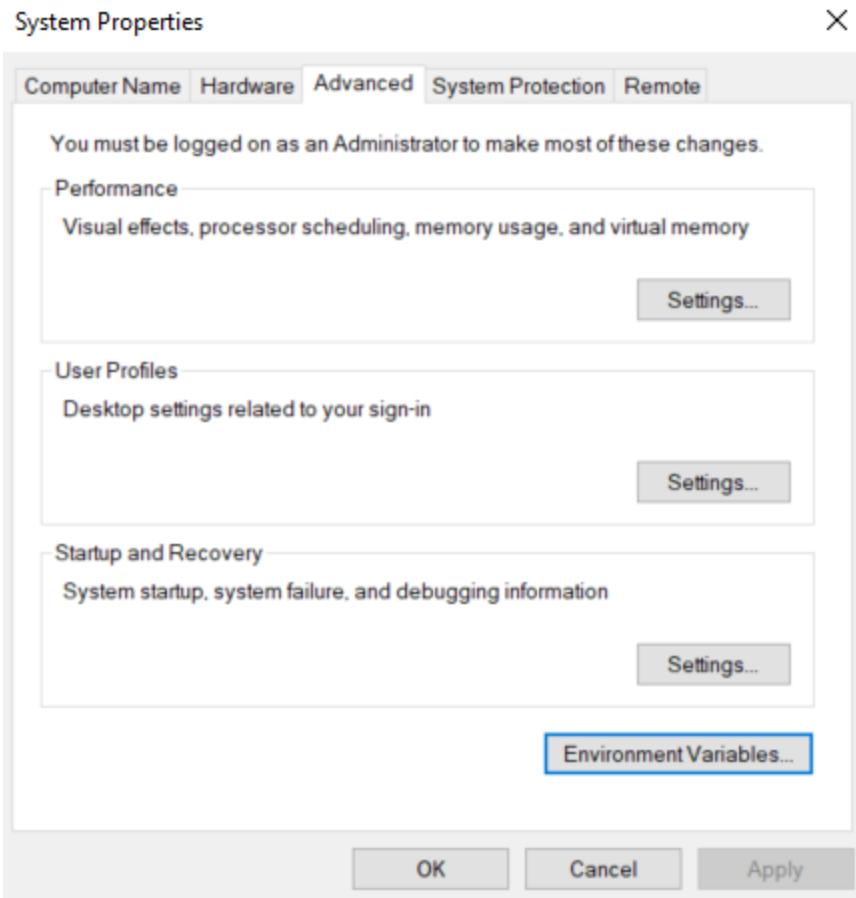
done
#
# To activate this environment, use
#
#     $ conda activate RAD2.0
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

Note: The main wizard GUI will be frozen while this process executes. Look at the background terminal window to check for progress, but do not close the wizard window as this will terminate the process and might affect future installation attempts.

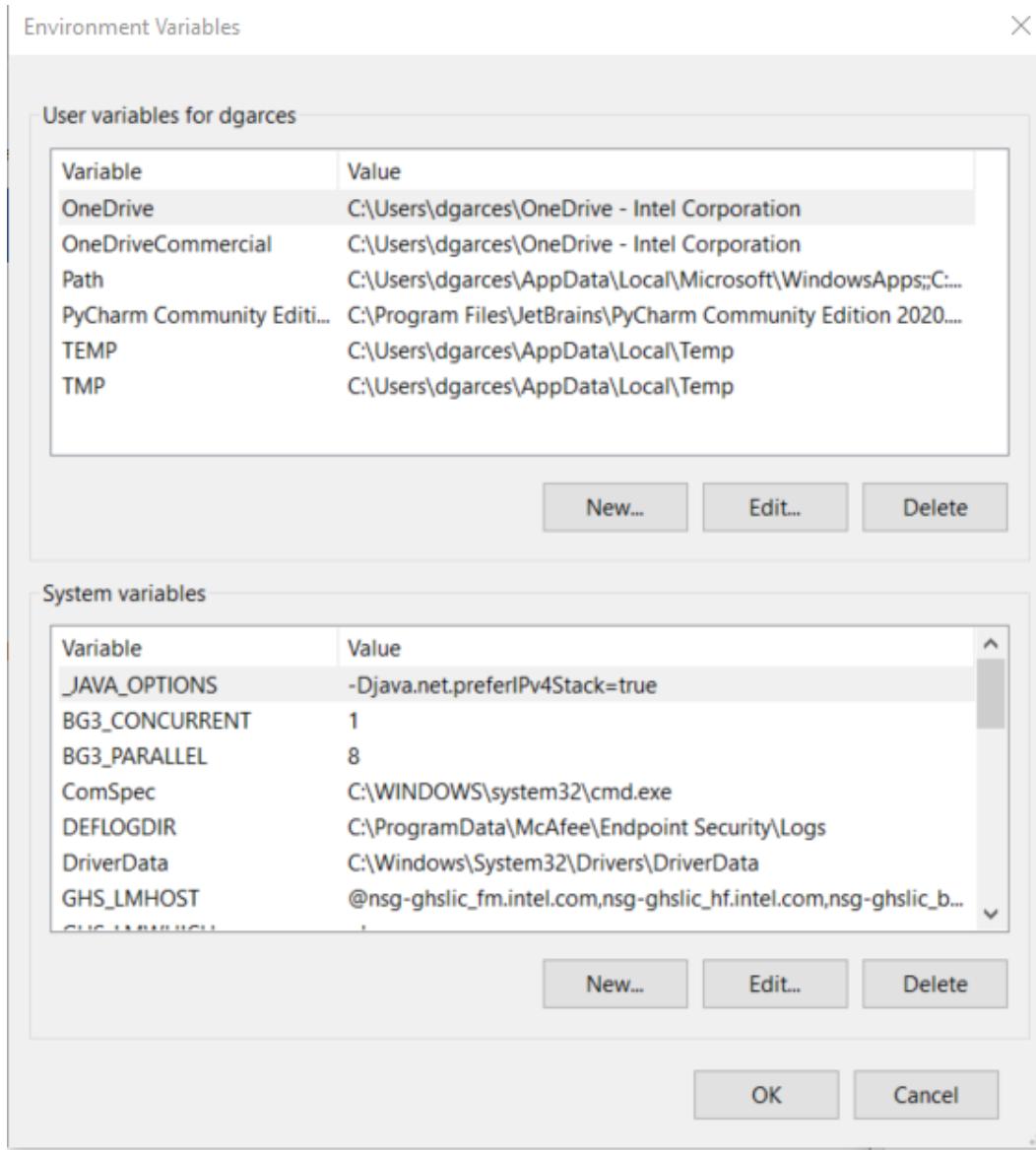
If you find a proxy error, please follow the following instructions, which are based on the initial solution proposed by Joe Tarango here: - Look for env in your search bar near the lower left of your screen. Go to "Edit the System Environment Variables" as shown below.



- Once this configuration opens, click "Yes" on the pop-up window.



- Click on "Environment Variables" as shown below.



- Click on the "New..." button for System variables and then add the following 3 variables as shown in the pictures below. Please consider that the examples below are for configuring the proxy with the US network. Change your configuration accordingly to reflect the location of the proxy that you want to utilize. Please refer to the code below for the easy cut and paste variable values.

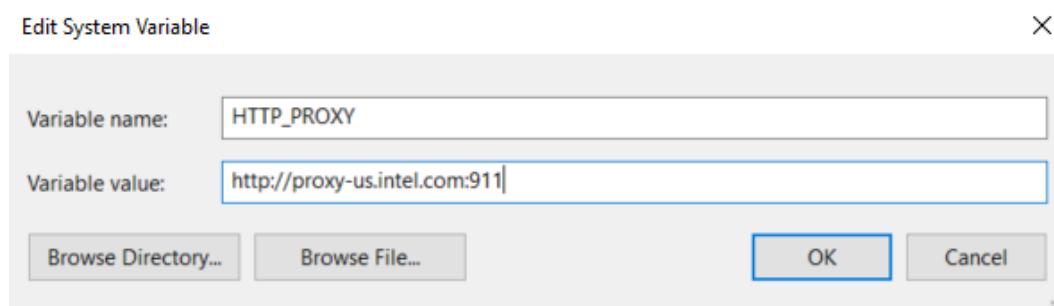
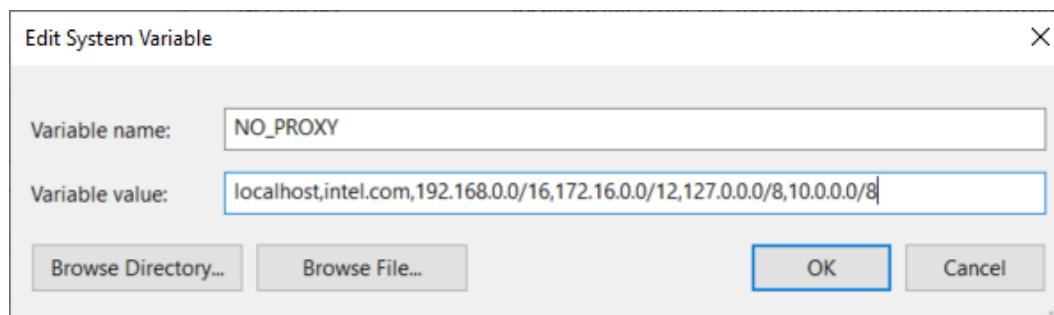
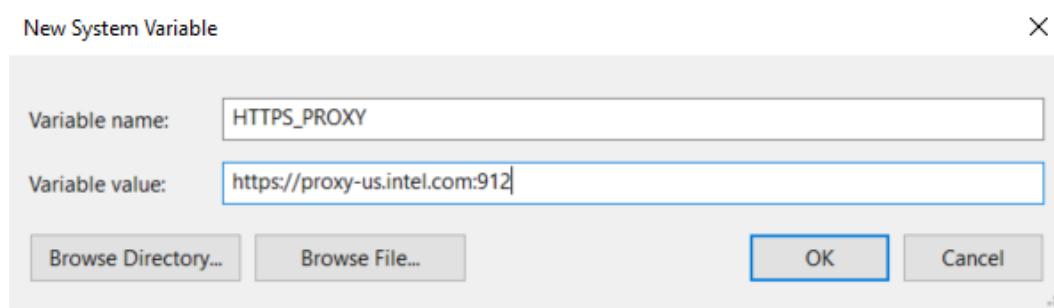
HTTPS_PROXY

https://proxy-chain.com:2

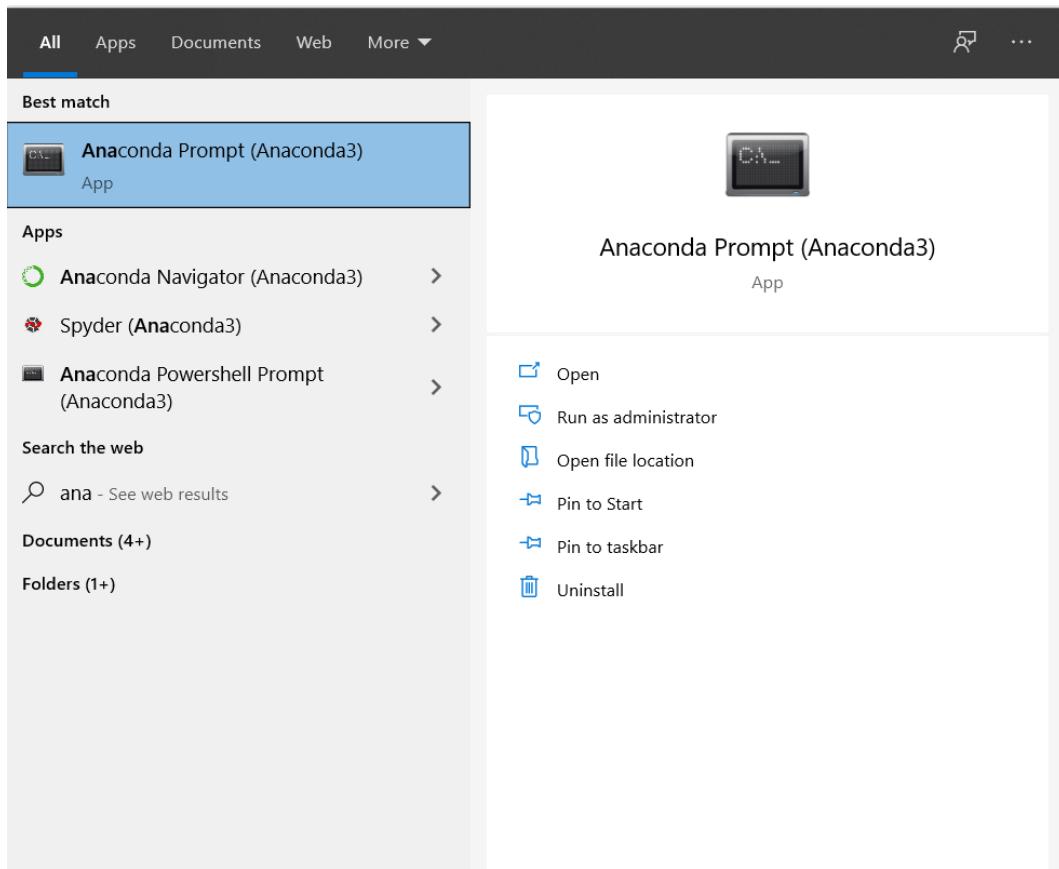
NO_PROXY

localhost,192.168.0.0/16,172.16.0.0/12,127.0.0.0/8,10.0.0.0/8

HTTP_PROXY
http://proxy-chain.com:1



- Open an Anaconda3 Prompt (it should have been installed during the Anaconda installation), and type: where .condarc.



Note: If there is no .condarc, please create this file inside C:Users<your-user>, where <your-user> corresponds to the user name that you are utilizing on your local machine.

```
(base) C:\Users\dgarces>where .condarc
C:\Users\dgarces\.condarc

(base) C:\Users\dgarces>
```

- Navigate to the file location and open the file with your editor of choice (emacs, vim, notepad++, etc). Modify your file to resemble the file shown below.

```
channels:
  - conda-forge
  - defaults
  - intel
  - pytorch
  - anaconda
  - bioconda
  - mkl

ssl_verify: true
allow_other_channels: true

# Proxy settings: http://[username]:[password]@[server]:[port]
proxy_servers:
http: http://proxy-chain.com:1
https: https://proxy-chain.com:2

# Implies always using the --yes option whenever asked to
proceed
always_yes: true

# Auto updating of dependencies
update_dependencies: true

# Environment variables to add configuration to control the
number of threads. Choose for you machine.
default_threads: 4

# Update conda automatically
auto_update_conda: true

# Enable certain features to be tracked by default.
track_features:
- mkl

# pip_interop_enabled (bool)
#   Allow the conda solver to interact with non-conda-installed
python packages.
pip_interop_enabled: true

# Show channel URLs when displaying what is going to be
downloaded.
show_channel_urls: true
```

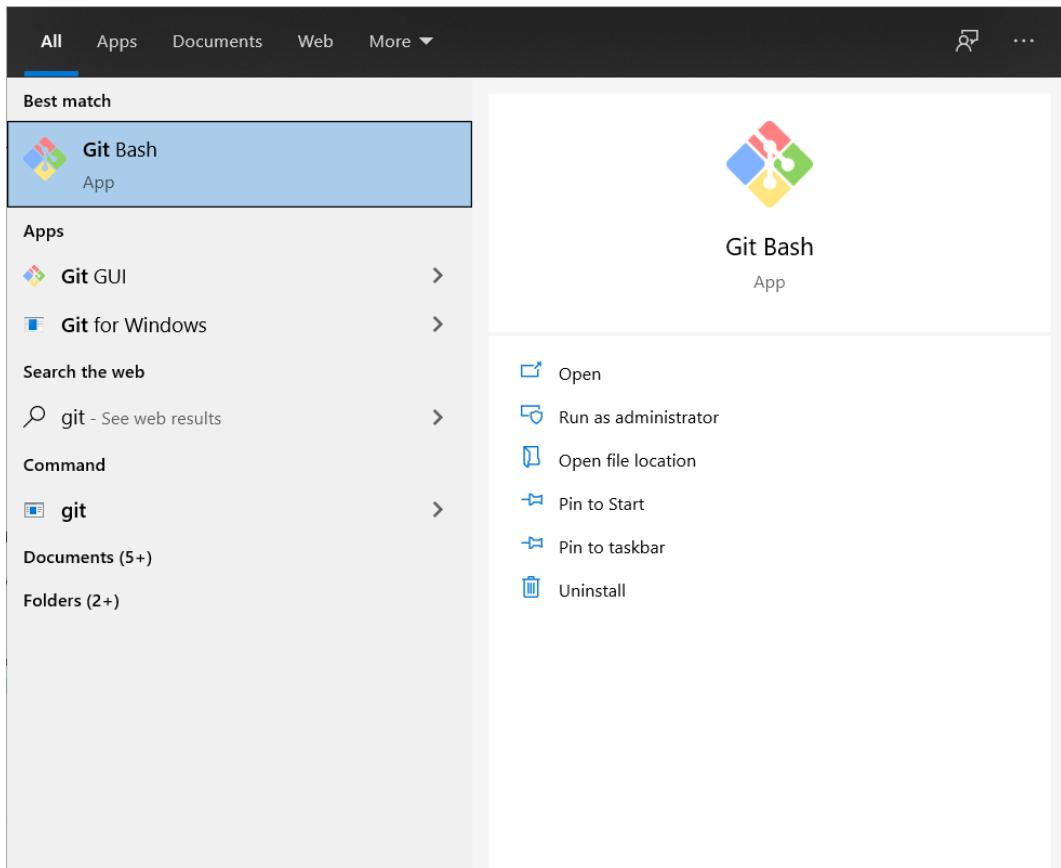
```
# Opt in, or opt out, of automatic error reporting to core
# maintainers.
# Error reports are anonymous, with only the error stack
# trace and information given by `conda info` being sent.
report_errors: false
```

- If you don't have it installed, please install the C++ Redistributable included in the Visual Studio 2019, which can be found here: <https://visualstudio.microsoft.com/downloads/>. After the download and the installation. Close all the installers and restart your computer. After your computer reboots, open the RAAD installation wizard again and try the environment creation again.
- If the issues persist, uninstall anaconda by following the instructions in <https://docs.anaconda.com/anaconda/install/uninstall/>, kill the RAAD installation Wizard and start again from step 1.
- If you face additional issues with anaconda not described in this tutorial, please use the following list to manually install the packages in the conda prompt. Navigate to Anaconda Prompt and execute each of the following commands independently. Wait for the command to complete its execution before running the next command

```
conda create --name raad python=3.8
conda activate raad
conda uninstall pip
conda install pip=20.2.4=py38_0
conda install wgetter
conda install pandas=1.0.5
conda install -c conda-forge scikit-learn
conda install statsmodels
conda install -c anaconda psutil
conda install -c conda-forge gputil
conda install -c anaconda cryptography
conda install -c anaconda pycrypto
conda install -c anaconda urllib3
conda install -c conda-forge atlassian-python-api
conda install -c conda-forge jira
conda install -c conda-forge pyinstaller
pip install matplotlib==3.3.1
pip install tensorflow==2.3.0
pip install gnupg
```

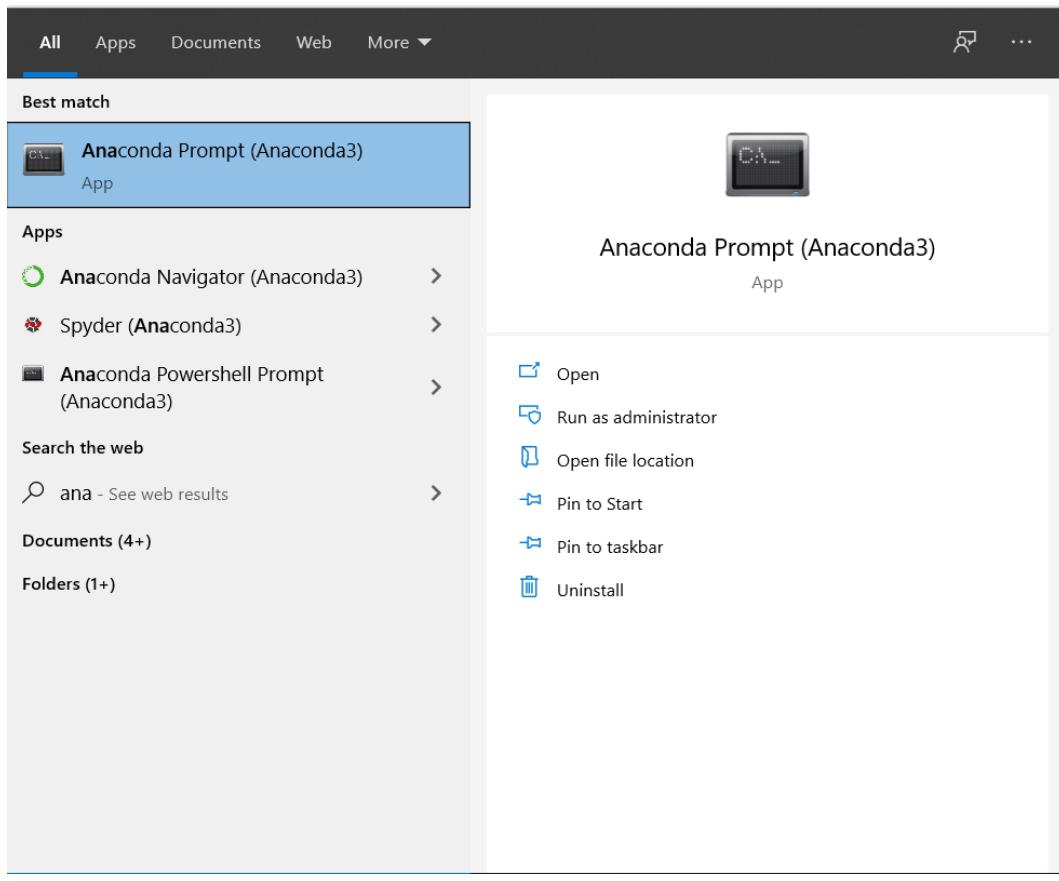
```
pip install PyLaTeX  
pip install tinyaes  
pip install PySimpleGUI  
pip install tornado  
pip install pycairo  
pip install unidecode  
pip install sentence-transformers
```

- After the creation of the environment finishes, you need to clone the RAAD repository by using git. If you do not have git, please go here to download it. Open a git terminal (as shown below), and execute the following command:

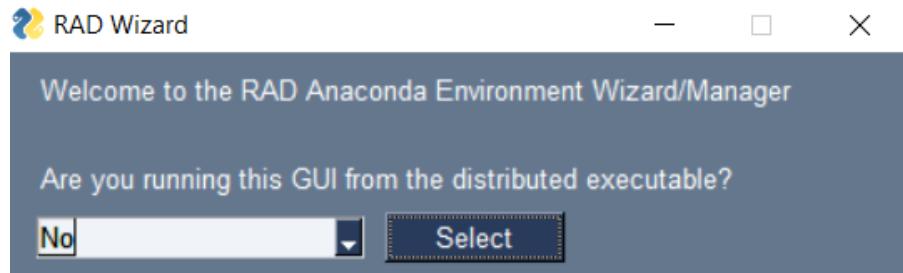


```
git clone https://github.com/Intel/RAAD.git
```

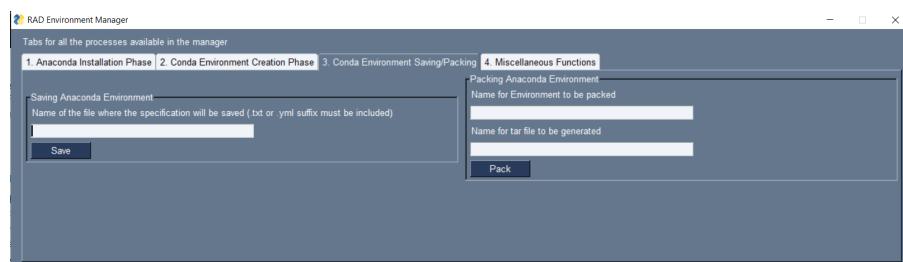
- After the repo has been cloned, close the installation wizard and the git bash terminal. Open an Anaconda3 Prompt (it should have been installed during the Anaconda installation) and activate the environment by executing conda activate RAAD2.0



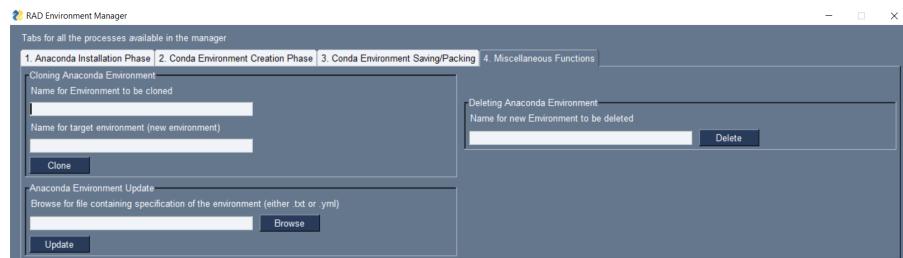
- Use the anaconda prompt to navigate to the location of the recently cloned "RAAD" repo and start developing using emac, vim, or your IDE of choice (I personally recommend PyCharm, which has a ton of useful features for developing in Python). If you decide to use PyCharm, you might need to open the repo using their interface, so it is properly initialized as a PyCharm project and you can use the embedded terminal for git and executing your scripts.
- If you want to easily manage your environments, navigate inside the repo to src/ and then run the installer GUI by utilizing the following command once you are inside src: python installer.py



- Select "No" in the first window for the installation wizard



- utilize the other miscellaneous functionalities that we have included to facilitate environment management for RAAD.



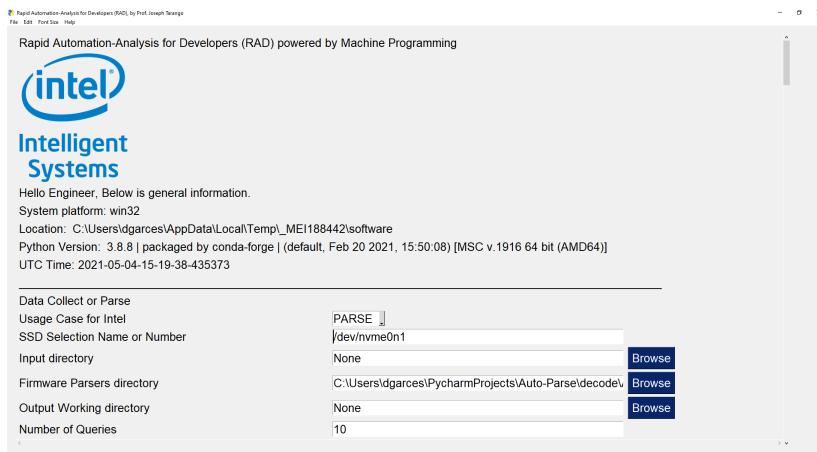
Information for Users of RAAD

As an user of RAAD, you need to access the GUI to run the functionality of our system,. There are two choices for running the GUI:

- Follow the instructions for developers all the way to steps above. Once you have a copy of the repo in your machine, run the main.py script by executing the following command from Anaconda3 prompt (with the right environment activated), or from your IDE of choice (as long as you are using the right environment for the interpreter) - please remember that you need

to navigate to src/ before running the command: python main.py --mode 1

- After around 30 seconds of automatic set-up, the system will generate a window like the one displayed below. Follow the subsection "Instructions for the GUI" for more information on how to operate the GUI.



- Create or use the GUI executable. @todo jdtarang folder inside the RAAD teams channel: The GUI is named gui.exe. Once you have the executable and its accompanying folder, make sure you save them both to the same location in your machine to allow the executable to operate correctly. The software file should contain a single file (the logo for the GUI), please make sure the logo is directly inside "software" and not nested inside other folders. To open and run the executable, just double-click on it.
 - You may encounter a blue window telling you "Windows protected your PC". Click on "More info" and then on the new button "Run Anyway"



- Follow the subsection "Instructions for the GUI" for more information on how to operate the GUI.



Note: Method 1 is preferable to guarantee that you get all the latest changes and recent code pushes that are constantly being added to the repo to improve the system functionality; however, method 2 provides a stable baseline with the basic functionality at the time of writing.

Docker Helpers

To use the docker scripts type:

```
cd raad/vm_ubuntu_22.04_x86_64  
source dockerHelp.sh
```

Show all docker containers running

dr-ps

Run specific command inside docker container without login into a container

syntax: cr-cmd <container> <command>

```
dr-cmd my-container ls -la
```

Run a docker image

```
dr-run <image>
```

Log in to specific docker container

```
dr-sh <container>
```

Show container logs for docker container

```
dr-log <container>
```

Build and run container on port 8080

```
dr-build <image:tag> <inside-port>
```

Reset docker container and its image

```
dr-reset
```

To run container that exited

```
dr-run-dead <image-name>
```

Reset/truncate docker container logs

```
dr-reset-log
```

Remove all unused images

dr-clean

Anaconda & Jetbrains Environment

- References
 - <https://docs.anaconda.com/anaconda/install/silent-mode/>
 - <https://www.jetbrains.com/help/pycharm/installation-guide.html#silent>
- Download binary for your operating system (Choose Anaconda3-2020+)
 - <https://repo.anaconda.com/archive/>

Linux Ubuntu Mate x64

Anaconda Installation

1. Download file

```
wget https://repo.anaconda.com/archive/Anaconda3-2020.07-Linux-x86_64.sh  
chmod 755 Anaconda3-2020.07-Linux-x86_64.sh  
. /Anaconda3-2020.07-Linux-x86_64.sh -b -p  
$HOME/anaconda3
```

2. Open Anaconda

3. Change to repository directory

4. Recreate and update the base environment with the best known method.

- Please note `environment_ubuntu-x86_64.yml` should be `environment_{operating system}.yml`

```
conda update --force conda -y  
conda update anaconda -y
```

```
conda update --all  
conda update anaconda-navigator  
conda update python  
conda env create --file environment_ubuntu-x86_64.yml
```

5. To update use

```
conda env update -f environment_ubuntu-x86_64.yml
```

JetBrains PyCharm Debugger

- <https://www.jetbrains.com/pycharm/>

```
sudo snap install pycharm-community --classic
```

Windows 10 x64

Installation

- Open command prompt

```
set url=https://repo.anaconda.com/archive/Anaconda3-  
2020.07-Windows-x86_64.exe  
set file=Anaconda3-2020.07-Windows-x86_64.exe  
certutil -urlcache -split -f %url% %file%  
start /wait "" Anaconda3-2020.07-Windows-x86_64.exe  
/InstallationType=JustMe /RegisterPython=0 /S  
/D=%UserProfile%\anaconda3  
start /wait ""  
%UserProfile%\anaconda3\Scripts\activate.bat
```

- Anaconda should be open or open Anaconda manually
- Change to repository directory
- Recreate and update the base environment with the best known method.
 - Please note environment_windows-x86_64.yml should be *environment_{operating system}.yml*

```
conda update --force conda -y
conda update anaconda -y
conda update --all
conda update anaconda-navigator
conda update python
conda env create --file environment_win-x86_64.yml
```

- To update use:

```
conda env update -f environment_win-x86_64.yml
```

Pycharm Debugger

Client Config via Command Line

- Create file: silent.config using create config file via command line

```
echo. > "silent.config"
echo mode=user >> "silent.config"
echo launcher32=0 >> "silent.config"
echo launcher64=1 >> "silent.config"
echo updatePATH=0 >> "silent.config"
echo jre32=1 >> "silent.config"
echo updateContextMenu=1 >> "silent.config"
echo python2=0 >> "silent.config"
echo python3=0 >> "silent.config"
echo regenerationSharedArchive=1 >> "silent.config"
echo .py=0 >> "silent.config"
```

- Optionally, edit config file manually:

```
; Installation mode. It can be user or admin.
; NOTE: for admin mode please use "Run as
Administrator" for command prompt to avoid UAC dialog
or user 'admin'.
mode=user

; Desktop shortcut for launchers
launcher32=0
launcher64=1
```

```

; Add launchers path to PATH env variable
updatePATH=0

; Download and install jre32. This may take a few
minutes.
jre32=1

; Add "Open Folder as Project" to context menu
updateContextMenu=1


; Download and install python. This may take a few
minutes.
python2=0
python3=0

; Regenerating the Shared Archive
; https://docs.oracle.com/en/java/javase/11/vm/class-
data-sharing.html
regenerationSharedArchive=1

; List of associations. To create an association change
value to 1.
.py=0

```

Commandline Install

- Installing in command line

```

set url=https://download.jetbrains.com/python/pycharm-
community-2021.2.2.exe
set file=pycharm-community-2021.2.2.exe
certutil -urlcache -split -f %url% %file%
start /wait "" pycharm-community-2021.2.2.exe /S
/CONFIG=.\\silent.config
/LOG=C:\\JetBrains\\PyCharmEdu\\install.log
/D=C:\\JetBrains\\Edu\\PyCharm_2020

```

Telemetry Focused Summary

Intel devices with the Telemetry system reduce the total cost of ownership associated with support services differentiating our ecosystem from competition. Our Telemetry process adheres to the NVMe 1.3 and SATA ACS 4 standard definitions; enabling customers (OEMs) to deploy a diverse portfolio of products with a common methodology to arbitrate a variety of devices in the field.

The two functionalities introduce are host-initiated telemetry asynchronous event notification (HiTaC) and controller-initiated telemetry asynchronous event notification (CiTaC). HiTAC allows the host platform to request persistent Metadata on-demand for monitoring operational feats such as: health, performance, quality of service, etc. The CiTaC enables the device to notify the host of OEM significant events requiring analysis to ensure product stability. The Meta data is dynamic and customized based on the operational events transpiring. These are further enabled by a novel auto decoding/translation of the Telemetry data sections without the requirement of proprietary and/or trade secret apparatuses such as internal tool chains. The procedure is so effective that the telemetry has absorbed and consolidated all data dumps within the system reducing complexity. The vision of telemetry is that the data provided alone can drive high confidence triage techniques to automatically locate, mark, characterize the fault, predict, detect anomalies, etc. Using our methods, the resources to resolve are significantly reduced, thereby focusing efforts on future products.

Mechanisms Architected for Success

The mechanisms within telemetry are architected so engineers can:

- Aid in characterization of the system setup, usage, and the sequence to the fault event.
- Diligently develop a problem statement using the scientific method by reducing scope.

- When a fault case is not clear provide feedback for how we can improve the methodology in future engagements.
- Improve the fault analysis handbook with encountered evaluations.
- Save all data critical data using available tools.
- Verify content.
- Provided concise list of telemetry data included.

Specifications Supported

Intel has decided to use the NVMe 1.3 Telemetry specification as a basis for all products (including SATA products based on ACS 4 definitions).

NVMe 1.3 Telemetry Specification

- Host Initiated Telemetry Log (log page identifier 0x07)
- Controller Initiated Telemetry Log (log page identifier 0x08).

SATA ACS 4 Definitions

- Current Device Internal Status Data Log (log address 0x24)
- Saved Device Internal Status Data Log (log address 0x25).

The both Telemetry specification defines that the page return data contains:

- Standard header specified - Data requested must be multiple of 512 Bytes
- Up to three consecutive data areas as defined by Intel. There are not part of the standard. - We defined what internal data needs to be packaged into the telemetry data sets. - The data list, values, and details are dynamic so the content can be customized per usage.

Telemetry Key Command Information

NVMe

- Identify controller data structure (ICDS; support of host and controller telemetry)
- Asynchronous event Configuration (AEC; host enablement of the (H/C)iTaC event notifications) feature identifier 0Bh
- Asynchronous event information notice (AEIN; detection field if telemetry log changed)
- Log page identifiers (LPI; 07h host, 08h controller)

- Create Host-Initiated Data (CHID; create a non-volatile snapshot of the telemetry data)
- Get log page (GLP; access information to the log page payloads)

SATA

- Identify controller data structure (ICDS; support of host and controller telemetry)
- Log Address(LA; 24h host initiated, 25h device initiated)
- Create Host-Initiated Data (CHID; create a non-volatile snapshot of the telemetry data)
- Read Log Ext (PIO) / Read Log DMA Ext (LA; access information to the log page payloads)

Summary of Specification Divergence

NVMe

- Telemetry is a non-blocking administration commands so other data command can occur.
- No other administration commands can be sent until the request is processed, per our transport architecture.
- Asynchronous event Request (AER) notification means the device can inform the host.
- The snapshot must be kept in a state value of 0x1. HI log per spec is regenerated anytime Create = 1 or there is a reset. The retain AER only applies to the CI log.

SATA ACS-4

- Telemetry is a blocking administration command so no other data can occur.
- No other command on the device can be sent until the command is processed.
- There is no AER (Asynchronous event Request) notification mechanism to inform the host if the drive has a Controller/ Device Initiated Telemetry (i.e. Event dump). Thus, the host has to request HIT or CIT header to see if data is available since the flags indicate the same content.
- The CIT log is cleared once the last block is read.

Type Meta Expectation

Controller Initiated Telemetry (CiTAC)

- Controller initiates a Get Log Page (NVMe 0x08, SATA 0x25) on internal minor or major issue is detected (such as: event dump or assert dump). Most other dynamic data is not current after a dump.
- Always provides a single data area (up to 31.9MB)

Host Initiated Telemetry (HiTAC)

- Host initiates a Telemetry data log file containing Data Areas (DA) 1-3 supported
- Controller initiated is driven by either an event or Assert; which will be in the HIT if pulled.

Generalized Cases

- System Fault Event Trigger Stop (General Case)
 - Software Assert, Event, and/or Hardware freeze.
 - Recommendation: Controller or Host Initiated generally get most of the data
- System Healthy with silent fault.
 - Detected host side verification fault
 - Recommendation: Trigger Event dump snapshot and Host Initiated
- Precondition: System Fault Trigger a stop for incompatibility
 - Recommendation: Trigger Event dump snapshot before update commit, then if fault occurs controller initiated
- System healthy with system at reduced potential resources. I.E. Garbage collection
 - Recommendation: Trigger Event dump snapshot, periodically trigger host initiated on data area 1 and 2 to get analytics including historical time series progression.

Customer/Developer Data Object Addition Process for Code

- Review Security Guidelines for Data Control
 - Determine if the data meets the guidelines
- Contact representatives
- Schedule Telemetry Working Group Meeting
- Presentation
 - Usage Development and/or Business requisite
 - Review Control/Data Flow (C/DFG) and Data of the Object Desired
 - Showcase the usage and impact
 - Provide urgency of deployment and stakeholders
 - Timeline for internal and/or external release
 - Domain Reviewers
- Allocate eUID to Sync Across products
- Verify telemetry swim lane functionality and auto generation parser functionality with trackable tidbits link
- Allocate Time
 - Live code review
 - Data extraction demonstration by walking through the C/DFG code path in Debugger
- Request Code Promotion
 - Once code pushed, dispatch notification to stakeholders

Access Telemetry using Ubuntu (Linux) NVMe-CLI

When wanting to access Telemetry packet in an open-source format as a single Application Programming Interface (API).

Summary

"binaryInterface.py" is a script that extracts uid-specific data from drive or binary, and returns it dynamically. It's a wrapper for other scripts such as "getTelemetry.py" and "parseTelemetry.py", but also implements nvme-cli library for Telemetry use in a Python format. Additional documentation on NVMe-CLI can be found in this Github below.

Environment

- You need to clone NVMe-CLI library
 - <https://github.com/linux-nvme/nvme-cli>
- This script needs to be run on a hardware-enabled environment. This meaning, most likely you want to run this code on a Test Machine. The test machine needs to be connected to the appropriate SSD drive you want to extract telemetry from.
- Note: device 0 is attached to NVME INTEL SSDDQXXX38T9 with name /dev/nvme0n1.
- These same numbers apply when telling the script which device to extract telemetry from, so take note.

Warning

Make sure you know the number of the device you are trying to access. Accessing device /dev/sg0, for example will fail as this is your local drive, and may sometimes have unforeseen consequences if written out to.

Step-by-step guide

Import tool code by cloning Github:

1. Import Open Source code by cloning Github Repo:
 - <https://github.com/linux-nvme/nvme-cli>
2. Create Directory
 - *mkdir ~/RAAD_Sandbox*
 - *cd ~/RAAD_Sandbox*
3. Clone
 - *git clone https://github.com/Intel/RAAD.git*
4. Navigate (cd) into script location
 - *cd ~/RAAD_Sandbox/RAAD/src/software/parse*
5. Import NVMe-CLI into same repo
 - *git clone https://github.com/linux-nvme/nvme-cli*
6. Run Script
 - Run script (nvme-cli -> parseTelemetryBin)

```
python
>> from binaryInterface import *
>> binCache, payload = hybridBinParse("5",
"/dev/nvme0n1")
>> payload = hybridChachedBinParse(binCache,
"6")
```

- OR (getTelemetry -> parseTelemetryBin process, normal)

```
python
>> from binaryInterface import *
>> binaryCache, payload = binParse("5")
>> payload = cachedBinParse(binaryCache,
"6")
```
- OR (getTelemetry -> parseTelemetryBin process, when you already have a binary file. NOTE: this version can be used without Test Machine)

```

python
>> from binaryInterface import *
>> binaryCache, payload = binParse("5", "
<NameOfSampleBinHere>")
>> payload = hybridChachedBinParse(binCache,
"6")

# To run a sample of tests demoing how to use this library

python binaryInterface.py

```

NOTE: see warning in Troubleshooting, scripts will need to be tweaked to be CL-usuable.

Output

You can expect the following output:

```

Phase 1 Extract
Data area 1 validity check pass!!!
Phase 2 Parse
Passed!!!
File Format: <Serial Number> <Core (Optional)> (eUID>
<Major> <Minor> <Known Firmware Name> <Byte Size>
Read Data Area 1 at Byte 512 of size 91136
Read Data Area 2 at Byte 512 of size 995840
Read Data Area 3 at Byte 1087488 of size 680960
uidInfoDict Contains the following UIDs:
['1', '2', ..., '5']
Printing Payload UID 5: dataOffset 24252, dataSize:
1024
Storing Payload in: PHA...5.1.0.bis.1024.1.0.bin ...
====objectId= 5, maj= 1, min= 0====
offset=24542
size=1024
name=bis
dataArea=1
rawData: BIS .....

```

Note: Printing of the binary data of the requested UID at the bottom.

Troubleshooting

NOTE: that this script still requires Command Line Inputs to be implemented, and the current version is hard-coded to produce some results for testing and learning purposes. This is because this library is made to be called in-code not used in CL, but can be tweaked to do so.

NVMe CLI Usage Commands on Linux Debug

Generic Commands

Example nvme command

```
sudo ./nvme -version
```

Get the identification

```
sudo ./nvme nvme intel id-ctrl /dev/nvme0
```

Extract Telemetry Payload

```
sudo ./nvme telemetry-log /dev/nvme0 -o telemetryPayload.bin  
-g 1
```

Clear APL max lifetime Temperature: A drive power-cycle or pci rescan may be needed.

```
sudo ./nvme admin-passthru /dev/nvme0 -o 0xF5 --cdw=x018  
sudo ./nvme subsystem-reset /dev/nvme0
```

Firmware Update

```
sudo ./nvme fw-download /dev/nvme0 -f firmware_file.bin  
sudo ./nvme fw-activate /dev/nvme0 -s 0 -a 1
```

Format 4K

```
sudo ./nvme format /dev/nvme0n1 -b 4k
```

Format 512B

```
sudo ./nvme format /dev/nvme0n1 -b 512
```

RAW Input-Output

To read the first LBA of a 4k formatted drive:

```
sudo ./nvme read /dev/nvme0n1 -s 0 -z 4096
```

It is recommended to pipe the output to another program to avoid raw binary output to the console (this example will print a hexdump of the block)

```
sudo ./nvme read /dev/nvme0n1 -s 0 -z 4096 | hexdump -C
```

To write to the first LBA of a 4k formatted drive. This assumes there is a file '4k.bin' available, this can be easily created using dd.

```
sudo ./nvme write /dev/nvme0n1 -s 0 -z 4096 -d 4k.bin
```

For random data:

```
dd if=/dev/urandom of=4k.bin bs=4096 count=1
```

For nonrandom data

```
dd if=/dev/zero of=4k.bin bs=4096 count=1'
```

Intel Specific

smart-log-add: Parses the Intel vendor unique SMART log page (0xCA)

```
sudo ./nvme intel smart-log-add /dev/nvmeX
```

market-name: Returns the marketing name for the drive, e.g. "Intel(R) SSD DC P4501 Series". (Log page 0xDD)

```
sudo ./nvme intel market-name /dev/nvmeX
```

temp-stats: Returns the temperature statistics for the drive. (Log page 0xC5)

```
sudo ./nvme intel temp-stats /dev/nvmeX
```

lat-stats: Returns IO latency statistics (Log pages 0xC1 for writes and 0xC2 for reads). Note this must be first be enabled with Set/Get feature 0xE2, e.g. 'nvme set-feature /dev/nvme0 -f 0xe2 -v 1'

```
sudo ./nvme intel lat-stats /dev/nvmeX
```

internal-log: Returns one of the event logs for the drive, (Example gathers the nLog)

```
sudo ./nvme intel internal-log /dev/nvmeX --namespace-id=1 -  
-log 0
```

Clear Assert (Opcode 0xD4): (NB) This will only work on specific firmware builds where the NCAT functionality is enabled (Usually Pre-PRQ, Amazon and Dell/EMC firmware), and only while the firmware is asserted. After this step a full reset of the drive is required (usually `warmReset()`) 'nvme reset /dev/nvmeX' is not sufficient. If in doubt do a full platform reboot.

```
sudo ./nvme admin-passthru /dev/nvmeX --p [cpde=0xD4 --  
cdw10=0x00 -w
```

Low Level Format (Opcode 0xD4): (NB) This will only work on specific firmware builds where the NCAT functionality is enabled (Usually Pre-PRQ, Amazon and Dell/EMC firmware), and only while the firmware is in a Bad Context state. After this step an 'nvme reset /dev/nvmeX' is usually sufficient to return the drive to a Healthy state.

```
sudo ./nvme admin-passthru /dev/nvmeX --p [cpde=0xD4 --  
cdw10=0x04 -w
```

Useful Linux Commands

Check for PCIe devices (PCIe Link) from both controller ID 0 and 1

```
lspci | grep Non
```

Check for NVMe devices (NVMe Driver Loaded) and namespaces within those devices

```
ls -al /dev/nvm* or lsblk
```

NVMe subsystem reset device and Rescan or Reboot Host: Prior to 4.4 kernel

- reboot Host Platform (Linux: 4.4+ kernel)

```
modprobe -r nvme, echo 1 > /sys/bus/pci/rescan,
modprobe nvme
```

Example Bash Script accessTelemetry.sh

```
HAMMERTIME=$(date +%Y.%m.%d.%H.%M.%S)
FILENAME=CDDP_$HAMMERTIME.bin
DIRECTORY=Telemetry_Logs
LOCATION=$DIRECTORY/$FILENAME
NVMTARGET=/dev/nvme0
CLIPATH=~/git/nvme-cli/
PWD=$(pwd)
cd $CLIPATH
echo ++++++
echo Checking All NVMe Devices...
echo ++++++
ls /dev/nvme*
echo ++++++
echo Checking All PCIe NVMe...
echo ++++++
sudo lspci | grep Non-Volatile
echo ++++++
echo Identify NVMe SSD at Index $NVMTARGET...
echo ++++++
sudo ./nvme id-ctrl $NVMTARGET
echo ++++++
echo Telemetry Get Host Log at Index $NVMTARGET...
echo ++++++
sudo ./nvme telemetry-log $NVMTARGET -o "$FILENAME" -g 1
chmod 777 "$FILENAME"

if [ -d "$DIRECTORY" ]; then
    # Control will enter here if $DIRECTORY exists.
    mv "$FILENAME" "$LOCATION"
else
    mkdir "$DIRECTORY"
    chmod -R 777 "$DIRECTORY"
fi
echo File at $LOCATION
ls -la $LOCATION
echo ++++++
cd $PWD
```

Usage Example to Log content

```
jtarango@Telemetry-Bench-0:~/git/nvme-clis$ sudo  
./accessTelemetry.sh  
+++++  
Checking All NVMe Devices...  
+++++  
/dev/nvme0  
+++++  
Checking All PCIe NVMe...  
+++++  
02:00.0 Non-Volatile memory controller: Intel Corporation  
Device 0d54  
+++++  
Identify NVMe SSD at Index /dev/nvme0...  
+++++  
NVME Identify Controller:  
vid      : 0x8086  
ssvid    : 0x8086  
sn       : BTLP82300JUP15PDGN  
mn       : INTEL SSDPD2KS150T8  
fr       : VDAAD453  
rab      : 4  
ieee     : 5cd2e4  
cmic     : 0x3  
mdts     : 5  
cntlid   : 0x1  
ver      : 0x10200  
rtd3r    : 0x989680  
rtd3e    : 0xe4e1c0  
oaes     : 0x100  
ctratt   : 0  
rrls     : 0  
cntrltype: 0  
fguid    :  
crdt1    : 0  
crdt2    : 0  
crdt3    : 0  
oacs     : 0x1f  
acl      : 127  
aerl     : 7  
frmw    : 0x2
```

lpa : 0xc
elpe : 127
npss : 0
avscC : 0
apsta : 0
wctemp : 343
cctemp : 353
mtfa : 0
hmpre : 0
hmmin : 0
tnvmcap : 0
unvmcap : 0
rpmbS : 0
edstt : 0
dsto : 0
fwug : 0
kas : 0
hctma : 0
mntmt : 0
mxtmt : 0
sanicap : 0x3
hmminds : 0
hmmaxd : 0
nsetidmax : 0
endgidmax : 0
anatt : 0
anacap : 0
anagrpmax : 0
nanagrpIid : 0
pels : 0
sqes : 0x66
cqes : 0x44
maxcmd : 0
nn : 0
oncs : 0x6e
fuses : 0
fna : 0x6
vwc : 0
awun : 0
awupf : 0
nvscC : 0
nwpc : 0
acwu : 0

```
sgls      : 0x70001
mnan     : 0
subnqn   :
ioccsz   : 0
iorcsz   : 0
icdoff   : 0
ctrattr   : 0
msdbd    : 0
ps      0 : mp:25.00W operational enlat:0 exlat:0 rrt:0 rrl:0
            rwt:0 rwl:0 idle_power:- active_power:-
+++++++++++++++++++++
Telemetry Get Host Log at Index /dev/nvme0...
+++++++++++++++++++++
File at Telemetry_Logs/CDDP_2020.01.17.19.33.35.bin
-rwxrwxrwx 1 root root 1208320 Jan 17 19:33
Telemetry_Logs/CDDP_2020.01.17.19.33.35.bin
+++++++++++++++++++++
```

References

1. <https://nvmeexpress.org/>
2. <https://www.mankier.com/1/nvme>
3. <https://github.com/nvme-compliance/dnvme>
4. <https://github.com/nvme-compliance/tnvme>
5. <https://github.com/linux-nvme/nvme-cli>
6. <https://github.com/clearlinux/telemetrics-backend>

Time Series

Tutorial

- <https://www.cs.ucr.edu/~eamonn/tutorials.html>

List of Rapid Learning based on Publications

- List of Publications and Links
 - Searching and Mining Trillions of Time Series Subsequences under Dynamic Time Warping SIGKDD 2012. Thanawin Rakthanmanon, Bilson Campana, Abdullah Mueen, Gustavo Batista, Brandon Westover, Qiang Zhu, Jesin Zakaria, Eamonn Keogh (2012). Best paper award [pdf].[data/code/video]
 - http://www.cs.ucr.edu/~eamonn/SIGKDD_trillion.pdf
 - <http://www.cs.ucr.edu/~eamonn/UCRsuite.html>
 - Instruction Set Extensions for Dynamic Time Warping. J. Tarango, E. Keogh, and P. Brisk. International Conference on Hardware/Software Codesign and System Synthesis (CODES-ISSS) Montreal, Canada, September 29 -October 4, 2013. [Paper] [Slides] [Poster]
 - <https://www.cs.ucr.edu/~jtarango/docs/codes13-edtw.pdf>
 - <https://www.cs.ucr.edu/~jtarango/docs/codes13-edtw.pptx>
 - <https://www.cs.ucr.edu/~jtarango/docs/grd.pdf>
 - Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View that Includes Motifs, Discords and Shapelets. Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, Eamonn Keogh (2016). IEEE ICDM 2016. [pdf] [slides]

- https://www.cs.ucr.edu/~eamonn/PID4481997_extrnd_Matrix%20Profile_I.pdf
- https://www.cs.ucr.edu/~eamonn/matrix_profile_i.pptx

Videos

- From Data to Knowledge - 102 by Dr. Keogh
 - <https://youtu.be/v5F-1K8GOYY>
- Shaplets, Motifs and Discords: A set of Primitives for Mining Massive Time Series and Image Archives by Dr. Keogh
 - https://youtu.be/sD-vvN_st58
- SAXually Explicit Images: Data Mining Large Shape Databases by Dr. Keogh
 - <https://youtu.be/vzPgHF7gcUQ>
- Searching and mining trillions of time series subsequences under dynamic time warping (KDD 2012) by Dr. Rakthanmanon
 - <https://youtu.be/xI26fQPbdX8>
- UCRSuite: Fast Subsequence Search (DNA)
 - <https://youtu.be/c7xz9pVr05Q>
- UCR Suite: Fast Nearest Neighbor Search (Top-1 NN)
 - https://youtu.be/d_qLzMmuVQg
- Data Mining Profiling Part 1 background by Dr. Keogh
 - <https://www.youtube.com/watch?v=1ZHW977t070>
- Data Mining Profiling Part 2 algorithms by Dr. Mudeen
 - <https://www.youtube.com/watch?v=LnQneYvg84M>
- Real time Motif Discovery on Parallel Accelerators by Dr. Zach Zimmerman
 - <https://youtu.be/fDmeCzpp96s>

Researchers

- Eamonn Keogh, Professor at University of California, Riverside.
- Abdullah Mudeen, Professor at University of New Mexico .
- Zach Zimmerman, Research Scientist at Google focusing on search.
- Philip Brisk, Professor at University of California, Riverside.
- Joseph Tarango, Adjunct Professor University of Colorado, Boulder.

Research Direction

- Problem statement
 - Time series data in the real world is multifaceted in nature and ordered. Algorithms in the past approximately matched queries, focused the solution on one data set, vary in computational cost, and storage space.
- Solution approach
 - Time series data in the real world are typically conserved in repeated patterns.
 - Searching through large sets of data we want to ensure all features are conserved.
 - Any measures should have high accuracy.
 - The methods should be simple and parameter free to prevent over-fitting.
 - The dimensionality of the query on a timeseries should be on the same order, meaning we want to maintain $O(N)$ or linear time.
 - Ensure missing data does not cause false positives.
 - The algorithm should be agnostic to the data set.
- Other state-of-the-art solutions
 - Spatial tuning through hashing, focused on one specific usage.
 - Similar algorithms increase in runtime polynomial as the query grows.

General API Background and Guidelines

Summary

The oneAPI programming model provides a comprehensive and unified portfolio of developer tools that can be used across data management engine, hardware targets, including a range of performance libraries spanning several workload domains. The libraries include functions custom-coded for each target architecture, so the same function call delivers optimized performance across supported architectures. Python, C, and C++ are the languages to based encourage ecosystem collaboration and innovation. To ensure comparability, we will want to have Cython and C/C++ APIs to connect to DPC++. The Intel wide APIs are defined in One API with the programming manual.

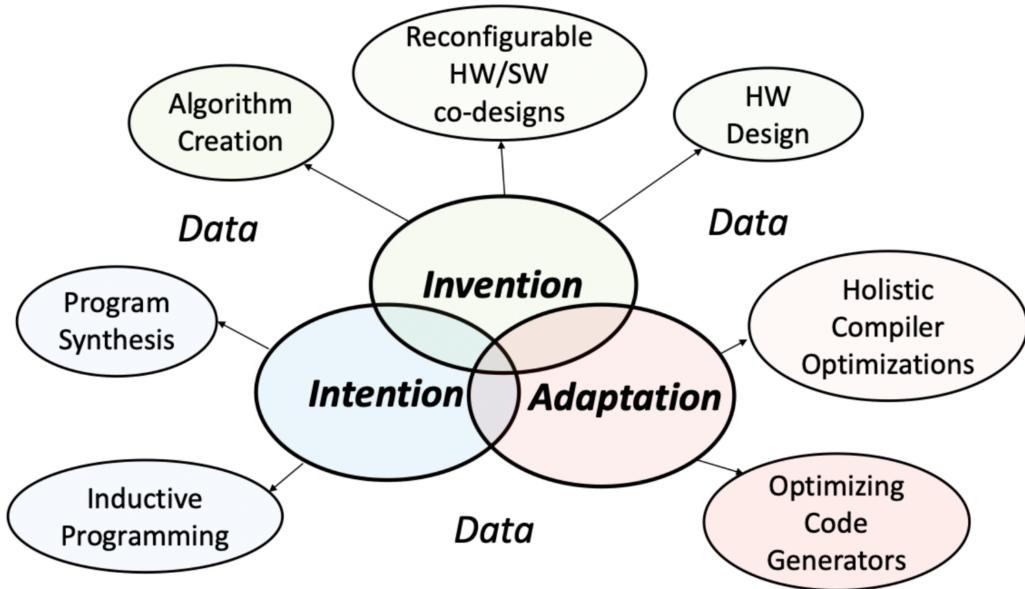
Within our development, we are embracing the Intel One direction and vision by collaborating on enabling Machine Programming. Machine programming is a fusion of different fields. It uses automatic programming technique, from precise (e.g., formal program synthesis) to probabilistic (e.g., differentiable programming) methods. It also uses and learns from everything we've built in hardware and software to date. Powered with OneAPI, we have a formula to cohesively have: onsite response, quality products, a service to improve up-time, and much more. Our usage has shown a 7x SLA speedup via automation of telemetry; thus, we are continuing the effort to scale organizations.

Machine Programming

<https://www.youtube-nocookie.com/embed/JMBEmUMSo8M>

Pillars

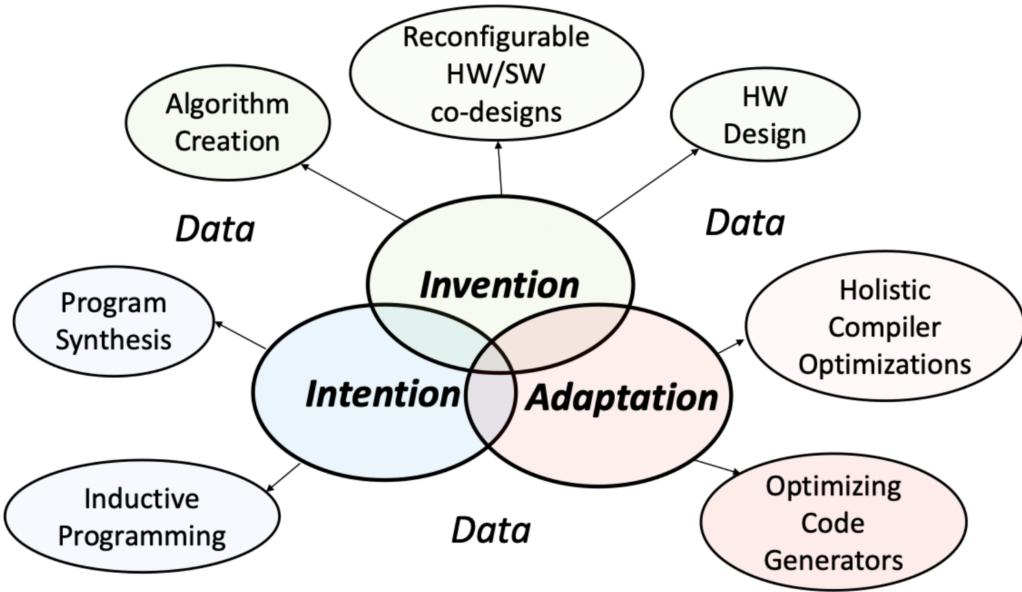
- Intention: Discover the intent of a programmer.
- Invention: Create new algorithms and data structures.
- Adaption: Evolve in a changing hardware/software world.



Three Pillars of Machine Programming.

High-level compelling endpoint general areas:

- Productivity: Improve programmer productivity by at least two orders of magnitude.
 - Requiring < 1% of today's effort.
- Synthesis: Superhuman software creation in terms of:
 - Performance, Correctness, Security, Accuracy
- Reconfiguration: Optimize operation through learning techniques by blurring lines of:
 - hardware, firmware, driver, and software.



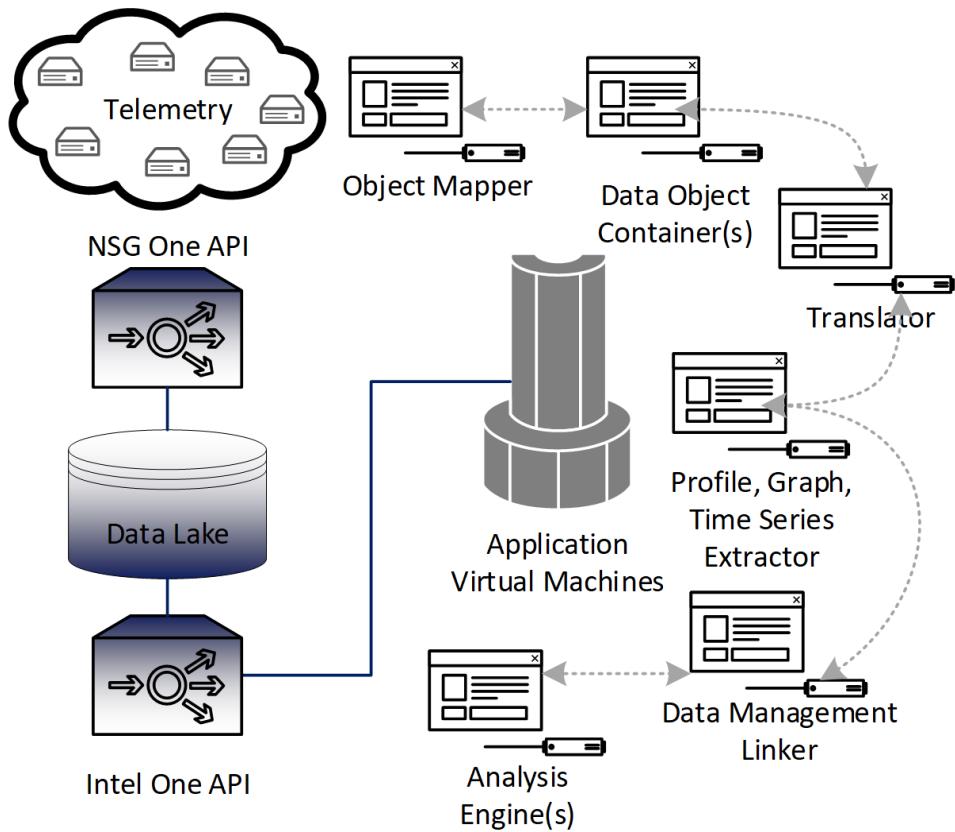
Scope of Machine Programming (MP).

Machine Programming Domains

- Compilers essentially transform higher-level algorithms and data structures to include lower-level details.
- Compute is hardware, algorithms, libraries, and ecosystem.
- Intelligence is focused on deep learning and machine learning.
- Synthesis is a program using probabilistic or direct transformation techniques.

Data Decomposition

1. Anomaly, healthy, etc. snapshot(s)
2. Binary
3. Map Identifier(s)
4. Data Object Container(s)
5. C, C++, or Python API
 - Directed Acyclic Graph
 - Time Series
 - Profile
6. Application Engine(s)



Application Instance(s) Topological hierarchy for the usage of data processing. Example instance for Business Unit (BU).

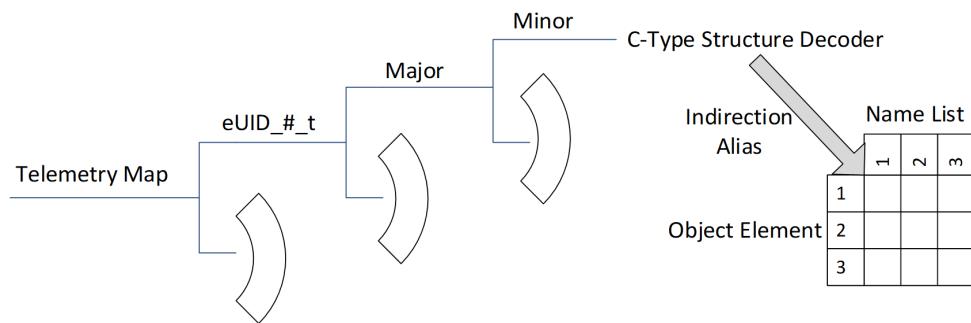
High Level Stages in the Application Instance(s)

- Object Mapper: API to map system, devices, products, etc. with the known features.
- Data Object Container(s): Translated objects within an encapsulated form.
- Translator: API to traverse the vector to the appropriate machine translator.
- Profile, Graph, Time Series Extractor: The APIs to machine programming generated signature paths, dependencies, relational information, etc.
- Data Management Linker: The data management engine to link all the Meta into the optimized topological instance for usage.
- Analysis Engine: Generic applications used to compute, infer, evaluate, etc. useful in rapid analysis. These can be stand alone or

chained operations to lead to conclusive findings.

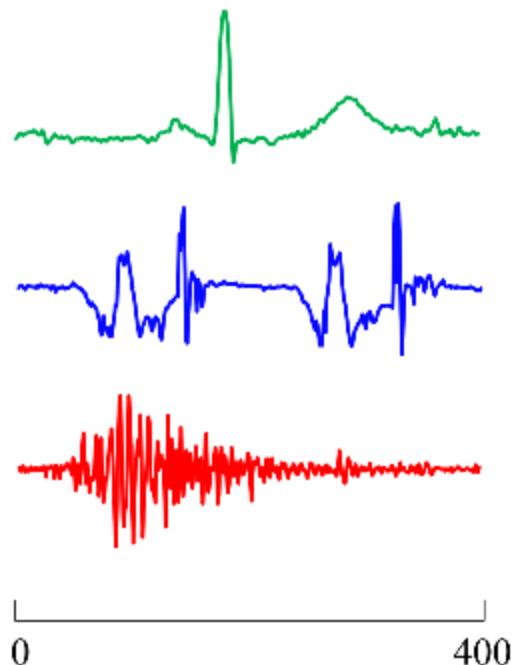
Dependency on Object Mapper to Data Object Container(s)

- eUID_#_t: Enumerated unique identification number of a given data object.
- Major: Version information representing a unique restructuring.
- Minor: Version information representing a unique restructuring.
- Data: The data payload with defined content.



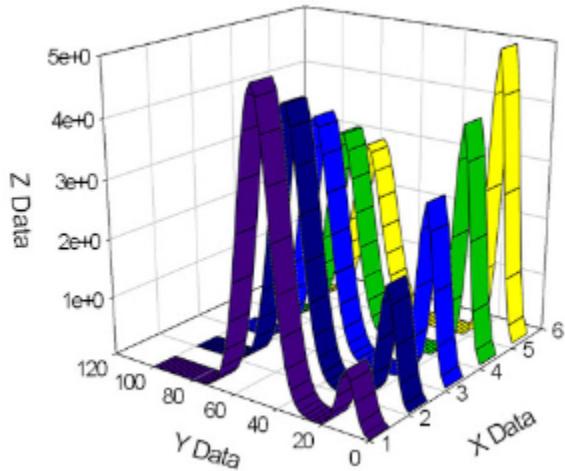
Data identification tree path for an individual structure.

Profile, Graph, Time Series Extractor



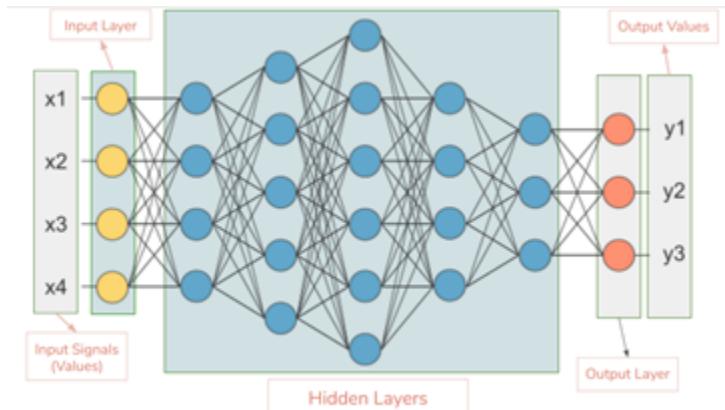
Example time series creation from data object container(s).

Data Management Linker



Example mapping of multiple time series in relational context.

Analysis Engine(s)



Neural network considering the features of a given data management link.

Data Save Requirement(s)

To adequately ensure we are encapsulating Meta data with reasonable context; we need to ensure the following Manifest List status is included below

Source System Context

- User Identification: External Customer (Dell, IBM); Internal (ConVal, RDT, EBT, Developer Bench); Parallel BU (DCG, CCG, PEG, Labs, etc.).
- Preconditioning: Sequential pack write of virtual Range.
- Previous Status: Device State space before execution context.
- Execution: Workload, Focused Feature Test, Platform Test, etc.
- Human Interaction: Connecting Probe, Pulling Parallel Devices (Hot-Plug), Firmware upgrade, etc.
- Machine Interaction: AI controlled Bot to simulate device hot plug every 10 seconds, control arm exposure to Alpha and Beta Particle generating radioactive material to simulate space, etc.

OS System Information

- EPOCH system time.
- Operating System with Update(s).
- Device Identification List, Status, up-time, Anomaly Time.
- References for System Information
 - <https://www.tecmint.com/commands-to-collect-system-and-hardware-information-in-linux/>
 - <https://docs.microsoft.com/en-us/powershell/scripting/samples/collecting-information-about-computers?view=powershell-7>

Solid State Drive (SSD)

- Data Payload(s): Binary format, time series linkage, Preconditioning.
 - Controller Initiated Telemetry Asynchronous Command (CiTAC).
 - Host Initiated Telemetry Asynchronous Command (HiTAC).

Compress API(s)

- Input
- Output
- Meta Context

- (Optional) Batch Requests

Relational Analysis Library API(s)

- Processing Module(s)
- Necessary Object Input(s)
- Intermediate Representation Object(s)
- Target Object Output(s)
- Dependency Graph
 - Input-Module(s)
 - Flow Map
 - List Generation
 - Projected
 - Resource(s)
 - Execution Time per Path
 - Power (Energy Consumed)

Developed Learning Path(s)

The constructed paths will be a certification to be an expert in the domain with necessary requirements.

References

1. NSG Data Lake specific AGS accesses (Inside Blue page).
<https://soco.intel.com/docs/DOC-2605534>
2. IT's Big Data Ecosystem with Tools, BKMs, Environments.
<https://wiki.ith.intel.com/display/BIBigData/Hadoop+Ecosystem+Tools>
3. Access Wiki for Intel Software Tools.
<https://wiki.ith.intel.com/display/ssgism/Parallel%20Studio%20tool%20page#ParallelStudioToolpage-GettingParallelStudio> and
<http://goto:ssg-ilc>
4. Justin Gottschlich, Armando Solar-Lezama, Nesime Tatbul, Michael Carbin, Martin Rinard, Regina Barzilay, Saman Amarasinghe, Joshua B. Tenenbaum, and Tim Mattson. 2018. The three pillars of machine programming. In Proceedings of the 2nd ACM SIGPLAN

International Workshop on Machine Learning and Programming Languages (MAPL 2018). Association for Computing Machinery, New York, NY, USA, 69–80.

DOI:<https://doi.org/10.1145/3211346.3211355>

5. "Why More Software Development Needs to Go to the Machines".
<https://newsroom.intel.com/news/why-more-software-development-needs-go-machines>
6. Machine Programming Direction Lecture (Justin).
<https://www.youtube.com/watch?v=JMBEmUMSo8M&feature=youtu.be>
7. Training Material <https://analyticsmarketplace.intel.com/>

Storage Container Web Interface and API Requirements to Data Lake Store

Creation of Web Server(s):

1. Reserve sub-domain
2. Create Apache Server instance linked with sub domain
3. Create a Web Application Firewall (WAF)
4. Create a TLS for Web interaction (Open SSL)
5. Add required analytics tracker for client webpage
6. Tracks information such as: MAC address, IP, Username, Geographic, Machine token identifier
7. Use interface such as:
 - Hardened PHP
 - http strict transport security (hsts)
8. Create isolated verification virtual machine instances
 - Instances check each of the items above. The server will not perform these tasks individually as it will be the first line of content verification.
 - Use a random of 3 multi casters virtual machines to simultaneously process the transaction in the case of a single attack vector.
9. Disable
 - Any additional ports

- Remote access to File upload server
- Write OS privileges for all users except service and stage data into containers.

10. APIs

- Python 3.x
- HTTP(s)/2 TLS
- Unique VPN DMZ for Telemetry

Security Requirement(s)

- Attack Protection Vectors
 - Entry point cannot execute binaries.
 - Each file should be scanned for malicious content.
 - File meta cannot be in plain text to ensure no file paths can be exploited.
 - Payload Size Limitation(s)
 - 1500 bytes of payload, the limit set by the IEEE 802.3 standard.
 - 9000 bytes of payload, the limit set by jumbo frames.
 - File Content Transaction(s)
 - Each set of content should have a SHA- 512 signature.
 - Data Encryption Public RSA Key 4096 bit.
 - Each data area section for construction of the payload we should have a SHA for a collection of blocks with a secondary private key only known to the tool.
 - Distributed denial of service (DDOS)
 - Connection instances are limited to the current usage of the server
 - For example if the server has 256 connections and only two users then the split would be 128. When more users connect these should be reduced to a minima of one connection with round Robbin Quality of Service (QoS).

- Cipher Point(s)
 - Key Exchange
 - Digital Signature
 - Message Authentication
 - Hashing Algorithm
- Connection to Server minimum TLS 1.3
 - TCP Port 443 is the standard port for HTTPS
- Key Set(s)
 - Public Certificate, SSID Certificate, Private 'Application Known' Certificate for User Signature

Application Programming Interface (API) for Standard Polymorphic Set-Container(s)

- Construction: Creation.
- Deconstruction: Deletion.
- Exception: Thrown exception for a catch handler.
 - Throw: Exception assertion and triggers for handling for graceful exit.
 - Catch: Exception de-assertion and trigger handler for object.
- Compress: Optimized the data structure or archival and space usage
- Decompress: Performs optimization for fastest run-time operation.
- Access: Methods used to gather features of the container.
 - Context: Data fields and relational information of the container.
 - Type: Inherent data type of underlying structure
 - Get: Gets a field or container set
 - Count (all, range): Counts the total features in a set or container.
 - Size: Total bytes occupied by container or subset.
 - Capacity: Total bytes possible for container before re-sizing.
 - Iterator: Creation of a pointer to the container for traversal.

- Empty: Status of the object context stating if there is no occupation of Meta.
 - Front: Iterator to the start of the container.
 - End-Back: Iterator to the end of the container.
 - Search (container, feature, relation): Operation of traversing object for given constraints.
 - Print: Streamer for console, text, python, CVS, JSON, etc. file output.
 - Permutations: Returns the possible total variants of the data container.
 - Combinations: Return the combinations of the variant container.
 - Copy: Returns a duplicate of the container.
 - Reference: Returns a unique reference hash to the container.
 - Generate Sequence: Generation functions of the container.
 - Sequence: Generates the sequence of operations on the object.
 - UML: Generates the graphical mapping of Sequence.
 - State Machine: Generates the traversal ordering of the sequence, actions, and transitions.
 - Adjacency list: List representation for a graph associates each vertex in the graph with the collection of its neighboring vertices or edges.
 - Translate (human, machine, type overlay): Unpacks a data container with the corresponding decoders; specifically, for machine formatted data.
- Mutate: Methods used to change the features.
 - Set: Changes a field or group according to usage.
 - Clear: Setting of all fields to a default or defined profile.
 - Erase (iterator,range): Erase the instance or range of instances.
 - Remove: Removal of the container from the data set and container returned.
 - Pop: Removes the data container set within the object.
 - Front: Removes the first data container set within the object.

- Back: Removes the last data container set within the object.
 - Push: Insertion a data into an object.
 - Front: Inserts the first data container set within the object.
 - Back: Inserts the last data container set within the object.
 - Insert (iterator, set): Insert of the data at a specific location within the object
 - Resize: Addition of space into the object in terms of bytes.
 - Operators: Basic mathematical, logic, Boolean, operators on data such as addition of integers or concatenation of strings.
 - Sort (compare): Sorts the container through compare rules.
 - Reverse (Compare): Sorts the container in reverse order.
 - Random Shuffle: Reorders the query into a random order.
 - Override: Mutates the structure into a destination container type or container reference.
- Map: Functions to perform high level tasks on container.
 - Unique: Removes duplicate copies of objects and links to single root reference.
 - Lower Bound: Returns the iterator for a given key.
 - Upper Bound: Returns the iterator before the given key.
 - Iterator Switch: Switches the iteration format based on traversing methodology.
 - Alias: Add reference to tag the entry with an additional tag.
 - Intersect: Determine the intersection or common container components
 - Union: Create the list of all features between containers.
 - Snapshot: Creates an instance of the container in the current form.
 - Profile: Creates a set of features identifiable for the type to use to overlay onto other data.
 - Similarity Measures: Algorithms to determine the feature similarity.

- Distance (Iterator(s)): Returns the unit distance between keys in vectored form.
 - Shortest Distance (Iterator(s)): Returns the optimized ordering of iterators to traverse.
 - Static Distance Measure: Measuring methods requiring exact context or a direct translator.
 - Euclidean Distance Measure: Direct 1 to 1 comparison.
 - Elastic Distance Measures: Methods with flexibility in measures such as Sequence Weighted Alignment Model being an unconstrained and constrained with Sakeo-Chiba or Itakura parallelogram.
 - Longest Common Sub-Sequence: Determines the least common subsequent between containers.
 - Dynamic Time Warping: Distance measures with repetitive patterns.
 - Edit Distance with Real Penalty: Distance measure with local time shifting.
 - Edit Distance on Real Sequence: Gap distance measure between objects with local time shifting.
- Stats: Enables tracking of the operations of the container.
 - Start: Enables feature.
 - Stop: Disables feature.
 - Difference: Return the change difference between two references.
 - Timer: Timer functionality feature enable.
 - Track: Calls and sequence traversing of the operations.
 - Basic: Triggers the enable of min, max, average, median, variation, deviation, etc.
 - Full: Triggers all of the features to be used.
 - Log: Log all of the operations conducted.
 - Regress: Constructs the minimum steps to reproduce a set of operations.

Library Infrastructure Interface Documentation

- MongoDB docs
 - <https://docs.mongodb.com/manual/>
- Mongodb docker
 - https://hub.docker.com/_/mongo
- Dockerised mongodb instance for reporting tool
 - Pymongo docs
 - <https://api.mongodb.com/python/3.6.0/tutorial.html>
- oneDNN performance reporting tool
 - https://gitlab.devtools.intel.com/ipl_infra/perf-reporter
- Flask docs
 - <https://flask.palletsprojects.com/en/1.1.x/>
- Atlassian
 - <https://blog.developer.atlassian.com/artificial-intelligence-for-issue-analytics-a-machine-learning-powered-jira-cloud-app/>

Getting Started with RAAD Tools

Instructions for the GUI:

The GUI has several sections at the time of writing, all of which will be covered below to help users more efficiently utilize RAAD. Remember that after running each section, the GUI will be frozen until the process finish running. To see the progress and potential warnings printed by each section, please refer to the background terminal window that pops up after the GUI executable is run. If when the GUI window launches, it does not display a sideways scrolling bar, please exit full-screen and re-enter it, so that the interface refreshes correctly and adjusts to your screen size and resolution.

Load and Probe Drive for Data Collection

Load and Probe Drive for Data Collection

Drive Workload Configuration	<input type="text" value="None"/> Browse
SSD Number	<input type="text" value="0"/>
SSD Name	<input type="text" value="/dev/nvme0n1"/>
Telemetry Pull Identifier	<input type="text" value="Tv2HiTAC"/>
Output directory	<input type="text" value="C:\Users\dgarces\PycharmProjects\StorageRelationalAr"/> Browse
Volume Label (Windows Specific)	<input type="text" value="PERFTEST"/>
Volume Allocation Unit (Windows Specific)	<input type="text" value="4096"/>
Volume File System (Windows Specific)	<input type="text" value="ntfs"/>
Volume Letter (Windows Specific)	<input type="text" value="G"/>
Partition Style (Windows Specific)	<input type="text" value="mbr"/>
Partition Drive (Windows Specific)?	<input checked="checked" type="checkbox"/>
Prep Drive (Linux Specific)?	<input checked="checked" type="checkbox"/>
Parse Binary Files?	<input type="checkbox"/>
Load and Probe Drive	

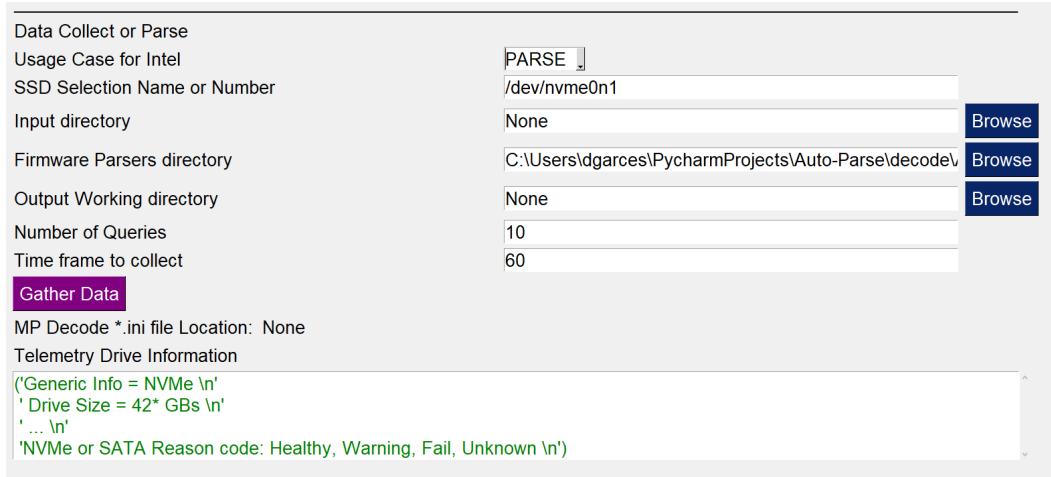
Important note: To run this section successfully, you need to execute the gui.exe as admin, so that IOMeter can operate successfully

This section of the GUI helps the user utilize a workload to load the specified drive before collecting telemetry data. For convenience, an additional option for parsing the collected telemetry data is added, so that

the user can obtain text files instead of binary files as the output of this section. The customizable inputs in this section are:

1. Drive Workload Configuration: path to the input file where the workload configuration is stored
2. SSD Number: Integer for the drive number from which to pull telemetry data
3. SSD Name: String for name of device interface to get data from
4. Telemetry Pull Identifier: String for the name of the data set that corresponds to the telemetry pull to be executed
5. Output Directory: path to the output directory where the binaries from the telemetry pull will be stored
6. Volume Label (Windows Specific): String for the label to be used on the disk volume
7. Volume Allocation Unit (Windows Specific): String for the volume allocation unit size
8. Volume File System (Windows Specific): String for the name of the file system to be used in the disk volume
9. Volume Letter (Windows Specific): String for the letter to be assigned to the disk volume
10. Partition Style (Windows Specific): String for the name of the partition style to be used in the specified disk
11. Partition Drive (Windows Specific): Flag to indicate if the program should partition the drive using the given parameters
12. Prep Drive (Linux Specific): Flag to indicate if the program should prep the drive before loading it
13. Parse Binary Files: Flag to parse the telemetry binaries pulled from the drive

Data Collect or Parse



This section of the GUI helps the user collect and parse binary files for telemetry. The resulting files after the parsing are text files containing all the information previously stored in the binary telemetry files. The customizable inputs in this section are:

1. **Usage Case for Intel:** This dropdown menu specifies the mode to be utilized. "PARSE" is for parsing previously collected telemetry data. "CLI", "IMAS", and "TWIDL" are all for collecting telemetry data from a specified SSD device. While "CLI" is the most general method for collecting telemetry, "IMAS" is an Intel specific external tool and "TWIDL" is an Intel specific internal tool. Bear this in mind when selecting a collecting method.
2. **SSD Selection Name or Number:** String specifying the name (path to the location) of the SSD device, or the number of the device on the TWIDL enabled memory list
3. **Input Directory:** Only used when "PARSE" option is selected. This input field specifies the path to the directory where the previously collected binary telemetry files are located.
4. **Firmware Parsers directory:** Input field specifying where the python parsing files for processing the binary files are located. These files are usually generated by Auto-Parse, so they are outside the current code repo. The code repo for Auto-parsers as well as an example telemetry binary can be found in Intel IMAS or NVMe-CLI releases
5. **Output Working Directory:** Input field specifying where the resulting text files will be stored.

6. Number of Queries: Number of telemetry pulls to be executed or parse depending on the option chosen for field 1
7. Time Frame to collect: Only used when "PARSE" is not selected. It specifies for how long should the system collect telemetry data. It serves as a time-limit to the execution of a large number telemetry pulls.

Fault Analysis Handbook Webpage (FAH)

Fault Analysis Handbook Webpage (FAH)

Username

Password

Loaded AES-256 Password Hash Signature

Status: Unknown web access...

Handbook Access

This sections of the GUI checks if the specified user has access to the Handbook to perform a crawl for information (to be utilized for failure prediction later). The customizable inputs in this section are:

1. Username: String specifying the username to access the handbook.
2. Password: String for the password associated with the username to access the handbook.
3. Loaded AES-256 Password Hash Signature: Generated signature for the specified password.

Telemetry Data Table

Telemetry Data Table

Decoded *.ini File **Browse**

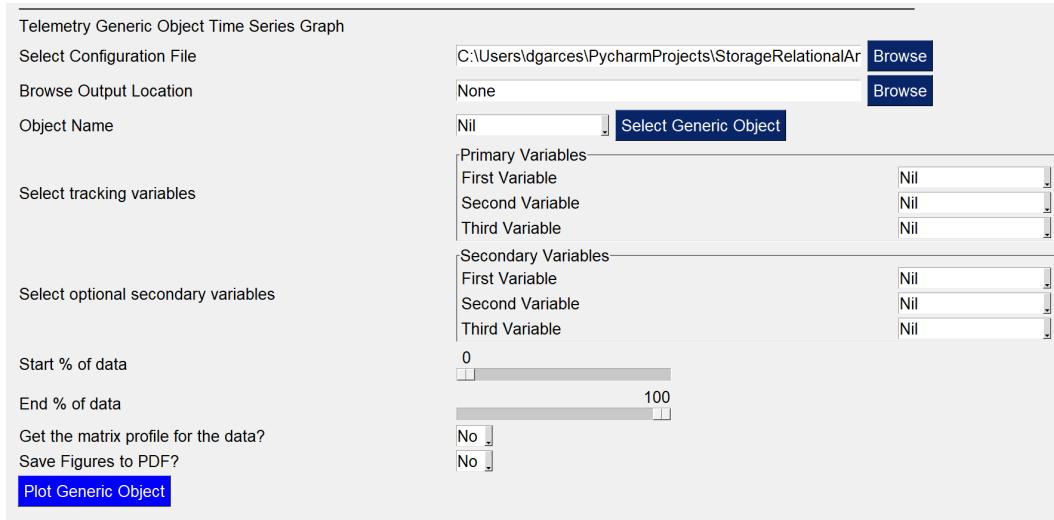
Choose Object to decode or choose all

Decode

This section of the GUI generates a data table to display the information contained in the decoded configuration (.ini) file. You can display the fields for a single object or for all the objects. The customizable inputs in this section are:

1. Decoded '.ini' File: Input to specify the path for the decoded configuration (.ini) file that contains the information to be included in the table.
2. Choose Object to decode or choose all: Drop-down menu containing all the object UIDs. To update the list of UIDs based on the specified '.ini' file, you need to click on the "Refresh Object UIDs" button.

Telemetry Generic Object Time Series Graph



This section of the GUI generates line graphs of the timeseries for different fields inside a single telemetry object. The customizable inputs in this section are:

1. Select Configuration File: Input to specify the path for the decoded configuration (.ini) file that contains the desired telemetry object to be graphed
2. Browse Output Location: Input to specify the path to the directory where the resulting PDFs will be stored - if "Save Figures to PDF?" is set to Yes
3. Object Name: String for the name of the object to be graphed
4. Select Tracking Variables: Object's fields to be graphed on the main axis (the y-scale on the left side of the graph)
5. Select Optional Secondary Variables: Object's fields to be graphed on the secondary axis (the y-scale on the right side of the

graph)

6. Start % of data: If set to a non-zero value, the first values of the time-series corresponding to the specified % are ignored when graphing all variables
7. End % of data: If set to other value that is not 100, the last values of the time-series corresponding to (1- specified %) are ignored when graphing all variables
8. Get the Matrix Profile for the Data?: Flag to indicate that the Matrix Profile of the data must be extracted before graphing it. Please refer to <https://www.cs.ucr.edu/~eamonn/MatrixProfile.html> to understand what Matrix Profile is.
9. Save Figures to PDF?: Flag to indicate whether the generated figures should be saved to PDFs or should be directly displayed to the screen for a single-use.

Telemetry Garbage (Defrag) Collection History

Telemetry Defrag History

Drive Type: CDR [Browse]

Browse Configuration File: C:\Users\dgarces\PycharmProjects\StorageRelationalAr [Browse]

Browse Output Location: None [Browse]

Select set points:

- Set Points:
 - start
 - normal
 - corner
 - urgent
 - critical

Primary Variables

First Variable	Nil
Second Variable	Nil
Third Variable	Nil

Select tracking variables

Select optional secondary variables

Secondary Variables

First Variable	Nil
Second Variable	Nil
Third Variable	Nil

Start % of data: 0

End % of data: 100

Is the secondary axis bandwidth?

Select the number of cores: No [2] [Yes]

Save Figures to PDF?: Yes

Decode Defrag Plot

Steps

1. This section of the GUI generates all the relevant graphs for analyzing Defrag History. The customizable inputs in this section are:
2. Drive Type: Drop down menu that allows the user to choose between two different drive types (CDR and ADP). Please select the drive from which the telemetry data values to be used were extracted.
3. Browse Configuration File: Input to specify the path for the decoded configuration (.ini) file that contains the desired telemetry data to be graphed. Please make sure this .ini file contains uid-41 as one of the objects.
4. Browse Output Location: Input to specify the path to the directory where the resulting PDFs will be stored - if "Save Figures to PDF?" is set to Yes
5. Select Set Points: Baseline reference lines used for evaluating whether HostWrites has fallen below each threshold. Each set-point represents a different threshold and associated drive state
6. Select Tracking Variables: Variables to be graphed in the main axis. We recommend you choose HostWrites and NandWrites if you are using the set-points, as these two fields will provide the best insight.
7. Select Optional Secondary Variables: Variables to be graphed in the secondary axis.
8. Start % of data: If set to a non-zero value, the first values of the time-series corresponding to the specified % are ignored when graphing all variables
9. End % of data: If set to other value that is not 100, the last values of the time-series corresponding to (1- specified %) are ignored when graphing all variables
10. Is the Secondary Axis Bandwidth?: Flag to indicate if the secondary variables will be used to calculate Bandwidth before graphing the resulting values
11. Select the Number of Cores: Select the number of cores in the Drive from which the telemetry data was extracted.
12. Save Figures to PDF?: Flag to indicate whether the generated figures should be saved to PDFs or should be directly displayed to the screen for a single-use.

ARMA Prediction Plot

The screenshot shows a user interface titled "ARMA Prediction Plot". It includes the following fields:

- "Browse Configuration File": A text input field containing "C:\Users\dgarces\PycharmProjects\StorageRelationalAr" with a "Browse" button next to it.
- "Object Name": A dropdown menu showing "Nil" with a "Select Object for ARMA" button next to it.
- "Select tracking axis variable": A dropdown menu showing "Primary Variable" with "Nil" selected.
- "Length of Window to be considered for Matrix Profile": A slider set to the value "8".
- "Get the matrix profile for the data?": A dropdown menu showing "No".
- A blue "Plot ARMA Prediction" button at the bottom left.

This section of the GUI takes a single object and a single field inside that telemetry object and then uses the Auto-Regressive Moving Average (ARMA) algorithm to predict future behavior of the field's values. The customizable inputs in this section are:

1. Browse Configuration File: Input to specify the path for the decoded configuration (.ini) file that contains the desired telemetry data to be used for the predictions
2. Object Name: Drop down menu listing all the objects contained in the configuration file. Choose the object that you want to use for the forecasting
3. Select Tracking Axis Variable: Object's field to be used in the forecasting
4. Length of Window to be Considered for Matrix Profile: Number of data values to be considered in a single window for Matrix Profile Extraction
5. Get the Matrix Profile for the Data?: Flag to indicate that the Matrix Profile of the data must be extracted before graphing it. Please refer to <https://www.cs.ucr.edu/~eamonn/MatrixProfile.html> to understand what Matrix Profile is.

RNN Prediction Plot

RNN Prediction Plot

Browse Configuration File C:\Users\ldgarces\PycharmProjects\StorageRelationalAI

Object Name Select Object for RNN

Select Field variables

Select Plot data

Input Width: 200

Label Width: 30

Shift: 30

Neurons Per Hidden Layer: 256

Batch Size: 64

Max Epochs: 2096

Categorical Encoding for Data? No Yes

Embedded Encoding for Data? No Yes

Optimizer for Neural Network: Adam

Activation for LSTM Layers: tanh

Activation for Dense Layer: sigmoid

Initializer for LSTM Layers: glorot_uniform

Initializer for Dense Layer: glorot_uniform

Apply Dropout Between Layers? Yes No

Get the matrix profile for the data? Yes No

Length of Window to be considered for Matrix Profile: 8

This section uses RNNs models to forecast the timeseries values for different fields of a single telemetry object. The customizable inputs in this section are:

1. Browse Configuration File: Input to specify the path for the decoded configuration (.ini) file that contains the desired telemetry data to be used for the predictions
2. Object Name: Drop down menu listing all the objects contained in the configuration file. Choose the object that you want to use for the forecasting
3. Select Field Variables: Object's fields to be used as inputs into the neural network. We have limited the number of fields to prevent users from generating bigger models that they would not be able to run locally.
4. Select Plot Data: For simplicity, we can only display a single field at a time. Use the drop down menu to choose the field to be

graphed. Remember that this field must also be part of the inputs to the neural network, or the GUI will not generate any graphs

5. Input Width: How big should the input time-series window be for forecasting future values. Bigger values will generate bigger models that require more computational resources, but allow for more accurate forecasting and a longer forecast output (we are able to predict further in the future)
6. Label Width: The number of time steps (data values) to be outputted by the neural network. This corresponds to the number of data values that will comprise the forecast.
7. Shift: How many data values should be skipped when shifting the input window to generate a new set of inputs for the neural network
8. Neurons Per Hidden Layer: How many units should be included in the LSTM and fully connected layers of the neural network
9. Batch Size: Number of data values to be considered before triggering a weight update in the neural network
10. Max Epochs: Number of iterations for training the neural network with the totality of the training set.
11. Categorical Encoding of the Data?: Flag that indicates whether the input is categorical (non-numerical), and therefore needs to be turned into a numerical value to be processed by the neural network
12. Embedded Encoding of the Data?: Flag that indicates whether the input should be encoded using complex embeddings. Usually recommended after generating a categorical encoding to uncover hidden relations between inputs
13. Optimizer for Neural Network: name for the optimizer to be used in the model. To know more about optimizers, please refer to:
<https://towardsdatascience.com/overview-of-various-optimizers-in-neural-networks-17c1be2df6d5>
14. Activation for LSTM layers: name of the activation function to be used in LSTM layers. To know more about activation functions, please refer to: https://en.wikipedia.org/wiki/Activation_function
15. Activation for Dense Layer: name of the activation function to be used in Dense layers. To know more about activation functions, please refer to: https://en.wikipedia.org/wiki/Activation_function

16. Initializer for LSTM layer: name of the weight initializer function to be used in LSTM layers. To know more about initialization function, please refer to:
<https://machinelearningmastery.com/weight-initialization-for-deep-learning-neural-networks/>
17. Initializer for Dense Layer: name of the weight initializer function to be used in Dense layers. To know more about initialization function, please refer to:
<https://machinelearningmastery.com/weight-initialization-for-deep-learning-neural-networks/>
18. Apply Dropout Between Layers?: Flag to indicate whether dropout should be applied between layers.
19. Get the Matrix Profile for the Data?: Flag to indicate that the Matrix Profile of the data must be extracted before graphing it. Please refer to
<https://www.cs.ucr.edu/~eamonn/MatrixProfile.html> to understand what Matrix Profile is.
20. Length of Window to be Considered for Matrix Profile: Number of data values to be considered in a single window for Matrix Profile Extraction
 - Note: The "Plot RNN Prediction" button will only generate a single graph at a time, so you might need to click it a total of 4 times to generate all graphs. Remember that you need to wait until the current graph is generated before you are able to click it again.

NLOG Predictor

NLOG Event Predictor	
Browse NLOG folder	<input type="text" value="C:\Users\dgarcés\PycharmProjects\StorageRelationalAr"/> <input type="button" value="Browse"/>
Browse NLOG Parser folder	<input type="text" value="C:\Users\dgarcés\PycharmProjects\StorageRelationalAr"/> <input type="button" value="Browse"/>
Number of Components	50 <input type="range" value="50"/> 50 <input type="button" value="▼"/>
Max Number of Parameters	8 <input type="range" value="8"/> 8 <input type="button" value="▼"/>
Input Size	4000 <input type="range" value="4000"/> 4000 <input type="button" value="▼"/>
Max Output Size	1000 <input type="range" value="1000"/> 1000 <input type="button" value="▼"/>
Model Type for Width Predictor	elastic <input type="button" value="▼"/>
Neurons Per Hidden Layer For Time Predictor	128 <input type="range" value="128"/> 128 <input type="button" value="▼"/>
Neurons Per Hidden Layer For Event Predictor	128 <input type="range" value="128"/> 128 <input type="button" value="▼"/>
Neurons Per Hidden Layer For Parameter Predictor	128 <input type="range" value="128"/> 128 <input type="button" value="▼"/>
Max Epochs For Time Predictor	2096 <input type="range" value="2096"/> 2096 <input type="button" value="▼"/>
Max Epochs For Event Predictor	2096 <input type="range" value="2096"/> 2096 <input type="button" value="▼"/>
Max Epochs For Parameter Predictor	2096 <input type="range" value="2096"/> 2096 <input type="button" value="▼"/>
Optimizer for Time Predictor	Adam <input type="button" value="▼"/>
Optimizer for Event Predictor	
Optimizer for Parameter Predictor	Adam <input type="button" value="▼"/>
Activation for LSTM Layers in Time Predictor	tanh <input type="button" value="▼"/>
Activation for LSTM Layers in Event Predictor	tanh <input type="button" value="▼"/>
Activation for LSTM Layers in Parameter Predictor	tanh <input type="button" value="▼"/>
Initializer for LSTM Layers in Time Predictor	glorot_uniform <input type="button" value="▼"/>
Initializer for LSTM Layers in Event Predictor	glorot_uniform <input type="button" value="▼"/>
Initializer for LSTM Layers in Parameter Predictor	glorot_uniform <input type="button" value="▼"/>
Apply Dropout Between Layers in Time Predictor?	Yes <input type="button" value="▼"/>
Apply Dropout Between Layers in Event Predictor?	Yes <input type="button" value="▼"/>
Apply Dropout Between Layers in Parameter Predictor?	Yes <input type="button" value="▼"/>
<input type="button" value="Execute NLOG Prediction"/>	

This section uses RNNs models to predict future NLOG events. The customizable inputs in this section are:

1. Browse NLOG folder: Path for the nlog Folder in which the nlog event files are contained
2. Browse NLOG parser folder: Path for the folder in which the NLogFormats.py script is contained
3. Number of Components: Integer for the number of dimensions to be used in the NLOG description embeddings
4. Max Number of Parameters: Integer for the maximum number of parameters that can be contained in an NLOG description for the specified formats file

5. Input Size: Integer for the number of NLOG events to be considered as the input for the predictive models
6. Max Output Size: Integer for the maximum number of NLOG events to be predicted with the models
7. Model Type for Width Predictor: name of the model type to be used in the linear regression model for determining the number of NLOG events to be predicted. Must be selected from the following: ['elastic', 'lasso', 'ridge', 'default']
8. Neurons Per Hidden Layer for Time Predictor: Integer for the number of neurons contained in each hidden layer for the NLOG time stamp predictor model
9. Neurons Per Hidden Layer for Event Predictor: Integer for the number of neurons contained in each hidden layer for the NLOG event predictor model
10. Neurons Per Hidden Layer for Parameter Predictor: Integer for the number of neurons contained in each hidden layer for the NLOG parameter predictor model
11. Max Epochs for Time Predictor: Integer for the maximum number of epochs to be considered when training the NLOG time stamp predictor model
12. Max Epochs for Event Predictor: Integer for the maximum number of epochs to be considered when training the NLOG event predictor model
13. Max Epochs for Parameter Predictor: Integer for the maximum number of epochs to be considered when training the NLOG parameter predictor model
14. Optimizer for Time Predictor: name of the optimizer to be used in the NLOG time stamp predictor model. Must be selected from the following: ['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax']
15. Optimizer for Event Predictor: name of the optimizer to be used in the NLOG event predictor model. Must be selected from the following: ['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax']
16. Optimizer for Parameter Predictor: name of the optimizer to be used in the NLOG parameter predictor model. Must be selected

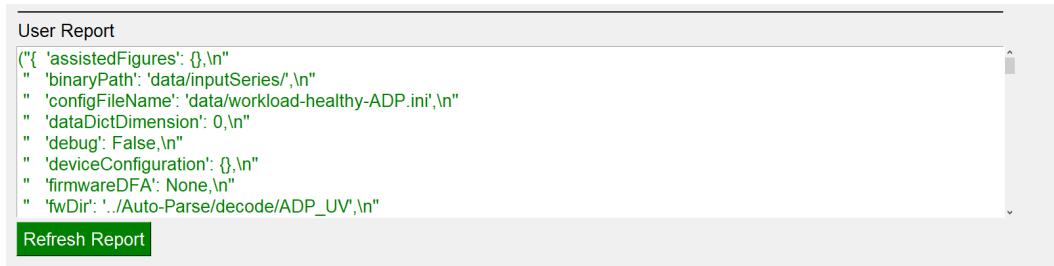
from the following: ['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax']

17. Activation for LSTM Layers in Time Predictor: name of the activation function to be used in the LSTM layers of the NLOG time stamp predictor model. Must be selected from the following: ['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']
18. Activation for LSTM Layers in Event Predictor: name of the activation function to be used in the LSTM layers of the NLOG event predictor model. Must be selected from the following: ['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']
19. Activation for LSTM Layers in Parameter Predictor: name of the activation function to be used in the LSTM layers of the NLOG parameter predictor model. Must be selected from the following: ['relu', 'sigmoid', 'softmax', 'softplus', 'softsign', 'tanh', 'selu', 'elu', 'exponential']
20. Initializer for LSTM Layers in Time Predictor: name of the weight initializer function to be used in the LSTM layers of the NLOG time stamp predictor model. Must be selected from the following: ['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']
21. Initializer for LSTM Layers in Event Predictor: name of the weight initializer function to be used in the LSTM layers of the NLOG event predictor model. Must be selected from the following: ['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal', 'glorot_uniform', 'identity', 'orthogonal', 'constant', 'variance_scaling']
22. Initializer for LSTM Layers in Parameter Predictor: name of the weight initializer function to be used in the LSTM layers of the NLOG parameter predictor model. Must be selected from the following: ['random_normal', 'random_uniform', 'truncated_normal', 'zeros', 'ones', 'glorot_normal',

```
'glorot_uniform', 'identity', 'orthogonal', 'constant',
'veariance_scaling']
```

23. Apply Dropout Between Layers in Time Predictor?: Boolean flag that indicates if dropout in between layers should be applied to the NLOG time stamp predictor model
24. Apply Dropout Between Layers in Event Predictor?: Boolean flag that indicates if dropout in between layers should be applied to the NLOG event predictor model
25. Apply Dropout Between Layers in Parameter Predictor?: Boolean flag that indicates if dropout in between layers should be applied to the NLOG parameter predictor model

User Report



The screenshot shows a window titled "User Report" containing a JSON configuration object. The object includes fields such as 'assistedFigures', 'binaryPath', 'configFileName', 'dataDictDimension', 'debug', 'deviceConfiguration', 'firmwareDFA', and 'fwDir'. A green button at the bottom left labeled "Refresh Report" is visible.

```
User Report
({
  'assistedFigures': {},
  'binaryPath': 'data/inputSeries/',
  'configFileName': 'data/workload-healthy-ADP.ini',
  'dataDictDimension': 0,
  'debug': false,
  'deviceConfiguration': {},
  'firmwareDFA': 'None',
  'fwDir': './Auto-Parse/decode/ADP_UV'
})
```

Refresh Report

This section will print the string representation of the User Report to be used for the first case in our Path Finding. Just hit Refresh Report to generate the report with the previously loaded data in section 1.

- Important Note: before you run this section, please download and install the MikTex distribution for Windows, so your system can correctly parse the output into a PDF. The MikTex distribution can be found here: <https://miktex.org/download>. After the download finishes, please install the MikTex distribution using the wizard (please make sure you choose the option to install it for all users in the computer, so that the script runs successfully). Also download the necessary packages as suggested in the installation wizard. If the MikTex distribution is unable to download the packages (known bug), please utilize an online compiler like Overleaf: <https://www.overleaf.com/> to compile the resulting .tex file.

Database Upload

The screenshot shows a web-based form titled "AXON Database Upload". It has a field labeled "Please enter your upload Destination and the file to upload" with a dropdown menu set to "test". Below it is a "Content File" field with "None" selected and a "Browse" button. At the bottom is a large green "Axon Upload" button.

This section will upload the zip file of the binary telemetry pulls to the Database. The customizable inputs in this section are:

- Please enter your upload Destination and the File to Upload: Drop down menu to specify the Axon location where the zip file should be uploaded
- Content File: Path to the zip file containing all the binary files to be stored

AXON Database Download

The screenshot shows a web-based form titled "AXON Database Download". It has a "Choose download directory" field with "C:\Users\dgarces\PycharmProjects\StorageRelationalAr" and a "Browse" button. Below it are fields for "AXON IDs", "Time created:", "Product Name:", "Serial Number:", and "User:". At the bottom is a large green "Axon Download" button. A "Download Information" section is visible below the buttons.

This section will download a zip file from the Axon Database containing the binary telemetry pulls. The customizable inputs in this section are:

1. Choose Download Directory: Path to the directory where the downloaded zip file will be stored locally
2. Axon IDs: Available objects to be downloaded. This are based on the User Profile specified in the next section

User Profile Information

User profile information	
Enter identity number	11487677
Enter username	jdtarang
Enter mode	gui
Key encrypt-decrypt location	None
Enable Encryption	False
Enter working directory	C:\Users\dgarces\PycharmProjects\StorageRelationalAr
<input type="button" value="Update Profile"/>	

This section allows the user to update their profile information by specifying a few parameters. The customizable inputs in this section are:

1. Enter Identity Number: Numerical value that identifies each user of RAAD
2. Enter Username: Username used for RAAD, the handbook connection, and Axon
3. Enter Mode: Mode of operation through which the user is accessing RAAD services
4. Key Encrypt-Decrypt Location: Path to the encryption key if one is available
5. Enable Encryption: Flag for encrypting communications and locally stored data
6. Enter Working Directory: Root directory from which the GUI is being run

Application Information

Application information for RAD	
Enter identity number	11487677
Enter major version number	1
Enter minor version number	0
Enter name	jdtarang
Execution location	C:\Users\dgarces\PycharmProjects\StorageRelationalAr
Enter mode	gui
Enter URL	http://www.intel.com
<input type="button" value="Update Applications"/>	

This section allows the user to update the application information by specifying different parameters. The customizable inputs in this section are:

1. Enter Identity Number: Numerical value that identifies each user of RAAD
2. Enter Major Version Number: New major version number to be assigned to the application
3. Enter Minor Version Number: New minor version number to be assigned to the application
4. Enter Name: Name of the developer that wants to request the changes
5. Execution Location: Path to the directory in which the root folder for StorageRelationalAnalysis is located
6. Enter Mode: Mode of operation through which the user is accessing RAAD services
7. Enter URL: URL for the web GUI

RAAD Extended Workloads

Several of RAAD's capabilities require time for training machine learning models. Learn how to setup a process to run RAAD overnight with the Linux screen command.

It is ideally best if you run RAAD on the high performance Linux server. Python Libraries (i.e., PyLaTeX) may or may not work properly on Windows. This tutorial is meant for running RAAD on the Linux server.

Server.rst

Step-by-step guide

1. RAAD may still be ran on Windows, though the following steps are no longer applicable. If it is your first time using RAAD and you plan on using Windows, please perform steps of the Information for Developers of RAAD section in Getting Started with RAAD Tools wiki to download RAAD and setup necessary Python environments.

- Please perform this step instead of step 7 in the Getting Started with RAAD Tools wiki mentioned above. After the creation of the environment finishes, you need to clone the RAAD repository by using git. If you do not have git please install it then open a terminal and execute the following command:

```
git clone --recursive https://github.com/Intel/RAAD.git
```

2. If it is your first time using RAAD on the a server:

- Begin by ensuring the grant of access to the server and to the necessary RAAD repositories.
- Use a terminal to execute the ssh command to log into your profile on the RAAD server.
- I.E. *ssh yourRADServerUserName@ServerName*

- Users may have to use Git Bash to login through ssh if you are remotely accessing the server from a Windows machine.
- Once you have accessed the server and logged into your profile, execute the following command to download all necessary repositories (shown below):

```
git clone --recursive https://github.com/Intel/RAAD.git
```

3. Create the file credentials.conf under the directory path

RAAD_Sandbox/RAAD/.raadProfile/

- Once this file has been created, use a text editor to add two lines: add your username to the first line of the file, followed by your password on the second line of the file. This file becomes a hidden credentials file used to access the Debug Handbook Wiki as well as the JIRA database. See the "fakecredentials.conf" file for an example.

4. If you have your own set/time-series of telemetry binaries you would like to process, do so by:

- First, renaming the original inputSeries/ directory within RAAD_Sandbox/RAAD/data/ to some other name, add your set of telemetry binaries into a new directory named inputSeries, and move that directory into the same data directory as the original inputSeries/ (i.e., move into RAAD_Sandbox/RAAD/data/)

5. After setting up the server and obtaining the RAAD repository:

- Connect to the server and log into your profile
- Enter the command: screen into the terminal to start a new screen session, then hit Enter again (shown below).
- Assuming you cloned the repository into your home directory on your RAAD server profile, enter the following command to change the correct directory for execution (shown below):

```
cd ~/RAAD/src
```

- The RAAD server should contain Python environments necessary for running RAAD. To activate, enter: conda activate RAAD

- Enter the command: *python main.py* to execute RAAD through one simple API (i.e., autoModuleAPI). Once RAAD begins to run, press "ctrl + a" on your keyboard, release, then press "d" once to detach from the screen session.
 - At this point, the execution will continue to run even after you log out of the server. You may now logout by entering *exit*.
 - At a later time, you can check the progress of the execution and/or observe the results and output after execution. Connect to the server once again and log into your profile (step 2 above). Then, to resume the previous screen session, enter *screen -r* (shown below).
 - Once you are finished with the session, you may enter the command: *exit* to terminate the screen session, and enter *exit* again to log out of the RAAD server.
6. By using screen, you may set up workloads to be run remotely in the background on the RAAD Server, and may resume the process which was used to execute RAAD at a later time.

Phases of Fault Analysis

Goal

Disposition customers challenges to determine the paths for mitigation to reduce exposure. Intention characterizes asserts, fault, warnings, etc. in a way that we can understand the impact such that we can decide if there are opportunities to for phases below.

Phases of Analysis

1. Document encountered challenge and preconditions.
2. Save meta data and state.
3. Analyze conditions and operational path.
4. Perform Scientific Process below.
5. Provide recommendations from analysis on how to close architecture challenges.
 - Eliminate the assert or failure
 - Recommending handling to recover
 - Create path to fail gracefully
 - Refactor the logic to narrow the failure conditions
6. Hypothesis of mitigation techniques.
7. Empirical mitigation techniques impact to reduction or clarification of fault.

Scientific Process

For first pass do not spend more than 20-30 minutes for evaluation. If the evaluation requires more information, seek out an expert or bring the gaps to the working group for acceleration of the item. Expectation is 1-3 other steps are a bonus.

1. Make Observations
 1. View the origin of the condition and trace the path to current state.

2. Understand and collect evidence, experiments, and correlation needed for usage. The process will involve understanding underlying mechanisms and system interactions to classify the failing case. The development of the failing case will involve understanding the control flow and data flow graph with the behavioral expectation to the error classification.
3. The general failure condition will show a trace of valid conditions until one is not met in the flow such that the coverage of all cases are depicted in a manner such as Boolean algebra, truth tables, sequence diagrams, finite state machine (deterministic finite automata), etc.
4. Clarify the focused experiment, cases, and predictions for the root cause. Gather data to conclusively replicate the failure or conditions that lead to such a situation.
5. Define the problem.

2. Think of Interesting questions

1. Understand necessary analysis data.
2. Identify paths for further investigation.
3. Create a strong correlation in the concerning area.

3. Formulate Hypotheses

1. The developer will cultivate and expand a hypothesis on the finding to give exact clarity to confirm defect based on analysis and usage of data.
2. Specify the requirements and identify variables violated in the defective case.
3. When resources or strategies are exhausted create a brain storming session or working group to further understand failure to develop a collective hypothesis or explore mechanisms until a relevant formulation is reached. In the event, of a hard failure note it so further investigation can continue.

4. Develop Testable Data and Predictions

1. Based on evidence, investigate the fault area beginning to formulate a predictive mechanism or existence of defect in design.
2. Refine, Alter, and Expand
 1. Upon proving the root cause of the failure, the developer will understand and propose solutions based on the architecture and implementation to address the concern and maintain future extensibility.
 2. Build a prototype to solution to test defective case proving solution definitively resolved the original defect.
 3. Before completing the solution, ensure the requirements are met and communicate results to technical lead, system technical product lead.
 4. The solution will follow the standards defined for the product library and appropriate reviews before promoting to live library testing.

5. Refine, Alter, and Expand, or Reject

1. After failure case resolution, communicate the resolution case where the solution will be tested in an independent controlled environment the failure was produced in confirming the solution is appropriate in the common product library.

6. Accept

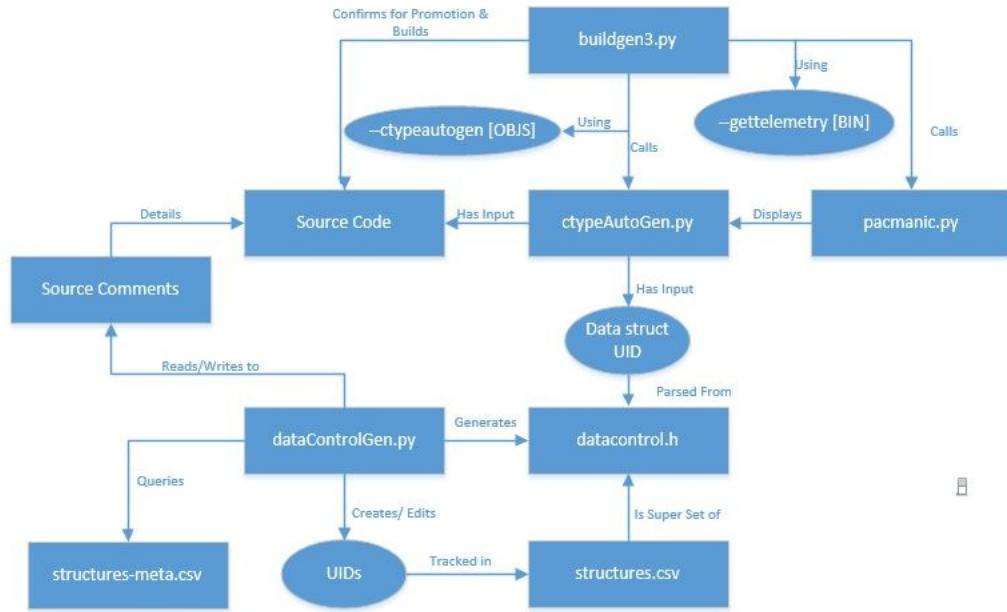
1. Solution is determined to be adequate for test case and corresponding horizontal technical leads should be made aware of the item resolution.
2. Follows to the solution that alter the architecture or expose failure conditions must be communicated to forward looking teams in a functional change in the ASIC, Firmware, Media, and or Tools.

Template of Documentation

- Title: <Assert/Fault/Unique Signature Code>
- Description: High level module information for the corresponding code and processes of dependant items.
- Evaluation Focal Points
 1. Evaluated Condition - Bullet list of conditions for event. -
Source code soft link
 2. Owner and Participant(s) - Developers and technical support members
 3. Seen at Customer (Yes/No) - Classification if internal or external encountered challenge
 4. Assert, Fault, Warning, Slient, etc. Condition Looseness or Ambiguity - Vagueness of the conditions around encountering event.
 5. Developer to do Analysis using “Scientific Process”
 6. Problem Statement
 7. Known causes
 - List of known classifications of causes
 8. Hardware/Firmware/Software Data Necessary for Analysis
 - List explicit annotated data structures
 9. Prevention recommendations
 10. Hardware-Firmware-Software Components
 - Links to code, architecture documents, versions, etc.
 11. Catastrophic Event leading to an Annual Failure Rate (AFR) post Reliability Demonstration Testing (RDT) (Yes/No) Why?
 12. Brown out or Power Loss Immanent (PLI) deadlock exposure (Yes/No) Why?
 13. Silent data error (SDE) or data miss-compare exposure (Yes/No) Why?
 14. Granularity of potential exposure to Silent Data Error Rate (SDER) (Yes/No) Why?
 15. Related items: <Assert/Fault/Unique Signature Code>
 16. Database items: Developer tracking system links

Data Control Assistance for Machine Programming

Our scripts manage reserving and updating datacontrol.h file upon creation of new data structures.



Datacontrol Flow

Related File Overview

The key feature of our automated management engines are each data structure is assigned a unique identifier (UID) to be tracked for the source code repository. The UIDs are key for identifying a data structure dependencies, classifications, etc. "findTags.py" is a script to detect data structures for source control management; which generates "baseTags.py" used as a baseline to detected aggregate data in source code. "cTypeAutoGen.py" and "autoParser.py" are variants of tools which uses input the project name and unique identifiers (UID[s]) of the data structures we would like to create parsers for it. The modes of operation are audit and generate, which will use defined objects and will attempt to audit/create parsers for all data structures it finds in the respective data control file (datacontrol.h).

"structures.csv" is the source which is used to generate "datacontrol.h" automatically and provide a reference for developers, system integrators, and validators. "datacontrol.h" is a C/C++ language target file used for the tracking and matching of UIDs to their structure names in the code repository. Depending on the repository and project for which it is compile/synthesized, "datacontrol.h" can contain different variants of structures supported in that code build. A feature of our management engine is "datacontrol.h" is a generated set of code semantics for developers to use. The purpose of code generation is to minimize human errors and ease requirement generation of a given data structure. It is recommended to use the provided interface since our system provides validation semantics on scripts and data formats such that in a rapid code release we can automatically update all stakeholders of the source code.

STEP 0: Guided Telemetry Input

Allows a guided walk-through of how to CREATE and EDIT data structures. This is important because it allows the detection of embeddings code documentation and standards directly into a UID creation/edit.

To get guided step-by-step of uid creation call:

```
$ python datacontrolgen.py --guide create  
0 error(s) found
```

To get guided step-by-step of uid editing call:

```
$ python datacontrolgen.py --guide edit  
0 error(s) found
```

STEP 1: Creating a Telemetry UID

Creation of a UID is started when the developer implements the structure in the native language source and uses Doxygen/Language style comments necessary in tagging a data structure. Each UID is processed by automatically specifying a temporary UID and assigning properties of the structure as done in the editing phase. Creating a temporary UID locally generated to the latest specification for the repository state generating the version control based on historical semantics making the process of updates an ease for developers.

To create a temporary UID, run:

```
$ python datacontrolgen.py --create [<temp-uid>,<structName>,<dataArea>,<dataGroup>,<product>,<editAssignments>]  
0 error(s) found
```

For Example:

```
$ python datacontrolgen.py --create [AUTO,testStructName1,6,Transport_PART,ADP,  
(description='This_is_a_description',owner='SomeUser',major='3',minor='10')]  
0 error(s) found  
Values in (<editAssignments>) need to be in format: variable='Non-space-alphanumeric-value' and  
multiple variables should be separated by a comma, making sure no spaces are introduced.
```

Multiple actions at Once

- Running multiple actions on the same command is supported, Each instance is separated by a semicolon (;) for scripting bulk modifications.

STEP 2: Editing a Telemetry UID

For a guide on which values can be specified, see "structures.csv" for examples, or Telemetry MetaData Guide below.

To edit a UID, specify the telemetry version as it appears on structures.csv that you would like to edit, then specify the option to edit:

- Edit can be called on both temp-UIDs and permanent UIDs. To edit run:

```
$ python datacontrolgen.py --edit [<temp-uid>,<telemetry_version>,(<editAssignments>)]  
0 error(s) found
```

The following command will result in Test_struct, with uid 33 for telemetry version 2.0, to be changed to Test_struct version 1.0 by Juan and respective others.

```
$ python datacontrolgen.py --edit [33,2.0,(version=1.0,owner=Juan Diaz)]  
0 error(s) found  
$ python datacontrolgen.py --edit [35,2.0,(version=1.0,owner=Joe Tarango)]  
0 error(s) found  
$ python datacontrolgen.py --edit [34,2.0,(version=1.0,owner=Andrea Chamorro)]  
0 error(s) found
```

Edits DO overwrite previous information, and for this reason for these changes to be permanent (exists outside of just your current working repository), they must be pushed by calling --implement

Edits will fail if you try to edit a temp_uid that does not exist and if you meant to create a UID from scratch with the edit properties, use the --create functionality.

STEP 3: Implementing a Telemetry UID(s)

Once you have a temporary UID, you can now call "--implement" to reserve a permanent UID for your structure. Until this is done, you will not have a permanent UID in the Application Programming Interface (API) used in the telemetry code, meaning your temporary UID will be generated with a different UID number each time "datacontrol.h" is generated. This process creates to enable testing and developing of temporary UIDs without reserving a permanent UID. The temporary UID will be generated and presented as permanent UID in the APIs used in the telemetry code, but keep in mind the UID number of those temporary UIDs can change each time "datacontrol.h" is generated. This process of implementing a UID ensures that no UID is assigned twice, maintaining uniqueness.

Temporary UIDs in structures.csv are in the format: temp_<identifier>

Temporary UIDs in datacontrol.h are in the same format as permanent UIDs.

To implement a uid: Create a temp_uid to track the Structure. Call:

```
$ python datacontrolgen.py --implement
Your struct temp-2 is assigned uid: 345
0 error(s) found
```

The call to implement will make changes in "structures.csv". These changes will govern a difference in "datacontrol.h", so regenerate "datacontrol.h" with:

```
$ python datacontrolgen.py --header
0 error(s) found

- Note: this generates a datacontrol.h header in the current directory. The build will automatically generate "datacontrol.h" as part of the build output artifacts on a per-build basis.
```

Commit your local changes to "structures.csv" and "datacontrol.h" onto your topic branch. It is important that "structures.csv" is updated properly; otherwise, you will have lost your newly made UID.

Implementations still need to be approved by a repo-meta superuser to be binding, but in the meantime ensure no one uses the UID you implemented by requesting it in our streamlined development tracking system.

STEP 4: Tagging a Struct in Source

Tagging occurs in source and is the foundation for automatic updating of "structures.csv".

To tag, Add the UID's data structure to the source:

```
struct newStruct_s {
//< UID: <description> and standard template information.
// singletons, data members, inherited objects, etc.
} newStruct_t;
```

Save the Source file.

STEP 5: Code Review

An repo-meta superuser needs to be added to the code review for your UID reservation change in order to approve the UID reservation on repo-meta. Adding data structures should have a valid reason and criteria for such that each is systematically architected, maintained, useful for debugging/logging. Please be aware the velocity of updates, and logging can impact performance so use our automated performance analysis tools to ensure changes do not impact critical paths unnecessarily.

STEP 6: Repo Pull

Request your branch code to be pulled to Trunk. Once pulled, data control reservation process is complete. Please remember we are a team of developers and using the process will ease communication and automate the boring stuff.

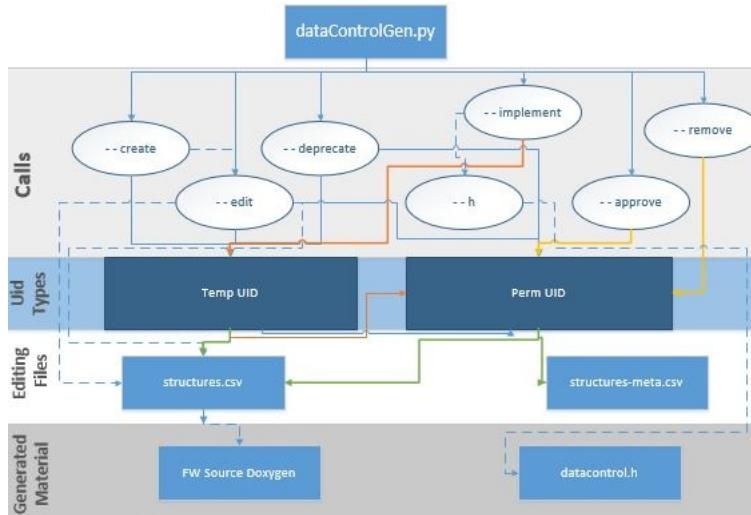
NOTES

Mitigating Incorrect Use of DatacontrolGen

- Periodic checking of repo-meta by superusers is the best way to ensure no damaging changes are made.
- The local "structures.csv" is the ultimate source of information for data control information specification in a repository, and incorrect changes will manifest themselves in the form or errors in running build scripts with flags such as --ctypeautogen.
- Automatic up keeping of "structures.csv" based on source code is a continuing process so improve source code when time permits in the development phases since small updates will limit large refactors.
- "structures.csv" should always be cross checked with repo-meta as part of the code review process.

Review Process

- UIDs are restricted to strings between 1 and 8 characters, and can only include \[a-zA-Z0-9\] characters and within the native language specification nomenclature.
- Due to language restrictions, no spaces are allowed when specifying list of UIDs in command line. THIS IS A PROBLEM FOR COMMENTS AND NAMING.
- "datacontrolGenMeta.py" is all or nothing, by design and will try to implement what it can.
- arguments to edit assignments that are meant to be read as strings and should be single quotes ('') since we use python as our preferred portable parsing language.
- Editing a specific version and UID is not supported since we have semantics in our model to ensure coherence and everything else can be edited (UID, MAJOR, AND MINOR LIKELY SHOULD NOT)
- If structure is novel a temporary UID allows for changes in the early development phase.
- If in debug mode, local repo-meta will not get committed or pushed
- For complicated inheritance, forward declarations, etc.; imports in "datacontrolGenMain.py" may fail due native language corner cases.
- Be careful that each --flag has a space before and after it as this is how we tokenize and identify disjoint elements.
- All Edits are only local and final edits occur automatically at the time of pushing to the tip branch and may require coordinated merging. Methods for automating these steps can be performed by our final push scripts since all final pushes are atomic in nature.
- Lastly, refrain from manual edits to "datacontrol.h" except for using "datacontrolgen.py" and if you see the need for improvements please do so with a stakeholder review. Remember we are a community a if you see room for improvement it can significantly improve organization velocity.



Flow for development.

Repo-META SUPERUSER COMMANDS

Approving Implementations

- You've called implement before the CSV file was ready to implement! Here are security measures to ensure things do not get messed up.
- Namely, the implementation still needs to be approved by someone who has access to repo-meta trunk. Approving can be done by a superuser using the "datacontrolGenMeta.py" script.
- To implement, checkout a local copy of repo-meta, and run:

```
$ python datacontrolGenMeta.py --approve <uid>
0 error(s) found
```

Push your changes to branch and if you don't have permission, an error will result.

Incorrect calls to --implement

- If an effort to create a new structure has been abandoned, get in contact with a superuser access to repo-meta, listed in contacts list in repo-meta, and make them aware an effort tied to the UID implementation has been abandoned. They will go in and --delete the UID, (only allowed if the UID has not yet been approved yet) to allow its future use. Otherwise, this UID has been reserved permanently.

Deleting Implementations

- If an effort to create a UID has been abandoned, all relevant UID information is removed from "structures.csv", and the UID has not been approved, a UID that was implemented can be reclaimed.
- Removing a UID that has been approved is not allowed and will result in an error.

A repo-meta superuser can remove the uid reservation in repo-meta by calling:

```
$ python datacontrolGenMeta.py --remove <uid>
0 error(s) found
```

Developers: Pushing Changes to Code Repo

- If developing in repo-meta, it is imperative to test your branch before pushing.
- To test use "datacontrolgenwrapper.py" in the repo
 - Edit it > Insert the following below repo.clone():
 - repo.gotoBranch("insertyourbranchnamehere")

This will prevent you from making pushes to origin in repo-meta when testing.

Code Review Process

"datacontrolGen.py" keeps a mixture of code-defined and non-code-defined characteristics, therefore, the review process for changes should be separated from the code review process for the code itself, so that these reviews can be directed and specific for the relevant reviewers. It is a good practice to limit code reviews with data-control related changes to firmware and metadata to their own branch, own code review for fast approval, and exception cases can be made for ninja developers (however, this is rare).

For this reason, to --implement changes to data control, a code review in repo-meta is created in its own branch to match your local changes, and code review of your local repo latest changes are made in parallel to a code review to your repo-meta changes. Only request pull to branch once both have been approved.

Data Control Struct Meta Information Grid Table

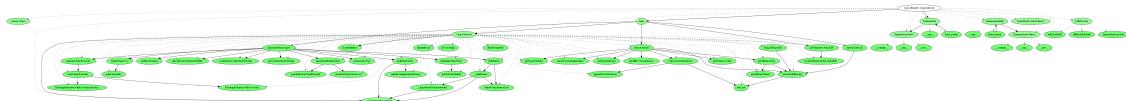
uniqueIdentifier	Telemetry Version	Autoparsable (Product, [Sub-Builds])	major	minor	persistence	dependency	Security Classification	byte size (Decim)

Definitions

- uniqueIdentifier
 - Documented unique value obtained by the --reserve process
- Telemetry Version
 - Prefilled to latest telemetry version and can be edited manually if need be. The interface of telemetry is version controlled by specifications.
- Autoparsable (Product, [Sub-Builds])
 - Documented by CtypeAutogen success, which updates the local csv. for a product and/or sub-builds
- Major
 - Major values represent novel changes and represents a change to backwards compatibility. These are automatically updated when source changes in synchronization with any commit to main repo. A local copy reflects currently working change generated by a source scan and pushed to remote branches for development context saves.
- Minor
 - A Minor value represents an addition to a data structure that has reserved bytes for desired changes. If the structure size change then a major version should occur with a reservation set of bytes for a product life cycle. These change are automatic updated and can be computable with all a specific major value.
- Persistence
 - Represents if a given value is volatile or non-volatile based on a given storage technology. Each persistence should be updated such that if critical/non-critical updates so for anomalous events the data can be rebuilt or recovered.
- Dependency
 - This is based on the inheritance of a given object.
- Security Classification

- Is a manual label given by a security expert and developer. The field is important to data leak audits and secret information either by the device or user.
- Byte size (Decimal)
 - Represents the base 10 mathematics size of a data structure from compilation or probing of source code.
- Data Area
 - Represents a data velocity, criticality, verboseness, etc. based on specifications. We label these to ensure updates are designed for the relative observation rates.
- Structure Duplication
 - Represents if there are more than one instance of a given type instance. This is useful to determine a specific instance such as update frequency between modules.
- Data Object Group
 - Annotation for labeling
- GlobalName
 - Useful for a global shared variable space between parallel operations.
- Size (Hex)
 - Hexadecimal base-8 size of a data structure.
- Within Assert
 - A tag to label the structure is in a event or assert dump payload.
- Product [Sub-Builds]
 - Attained by the product, build, or script. I.E. ctypeautogen and pythonicpy runs.
- Domain
 - Module or component of architecture/design.
- Owner
 - Determined by implementer "--implementor", can be manually edited. We track this so change search is relatively fast.
- Definitions
 - Represent the context of a given data structure. We use the definition to describe the purpose, usage, and features documented in the source code. It is determined at --implement phase.

How DataControlGen Works



Call graph

- List of initial data control structures can be found in a product review document and planning phase, if this does not exist then the source should be reflect present and future changes. This is the source of all data structures tracked by telemetry in the code.
- Use "getctypeautogen.py" to extract which new data structures have been added to firmware and are not matched to previous UIDs in database. An add requires appending and sorting the "New Structure List" along with the respective matching UID's to check for implementation conflicts.
- Changes between one version of the source code and the next version are recorded in repository management engine (I.E. Git) changes and only these changes would need to be parsed by "datacontrolGen.py" to determine a "New Structure List". Usage is similar to "ctypeautogen.py" or "autoparse.py" logic, which matches a UID to telemetry structure list and failed additions.

- A local copy of database is created upon calling "datacontrolGen", which makes calling "datacontrolGen" a pre-requisite of calling "ctypeautogen.py". Each change should be integrated into build after validation scripts.
- A build conflict happens if a structure and its UID is present in New Structure List and not implemented.
- If an object is implemented but not found by new data structure scan, our validation flow will throw a warning; resulting in no changes pushed to remote database.
- If a data structure has been implemented, and then deprecated, "Deprecate" its UID. UID info will be kept in datacontrol database as a commented deprecated section for future reference.
- If it was only "Reserved" but not Implemented , one can "Deprecate" a reservation, and it will not be kept track of. It is significant to note only an owner can "Deprecate" a reservation. Our tracking development system request is automated if "Deprecate" is called by non-owner and "Ask owner to Deprecate" confirm then redirects to automated development system.
- Implementation Conflict occurs if a user tries to implement a UID that has not previously been Reserved. Reservation can only happen if not implemented or a slot is reserved in UID database.
- "Free" UIDs are determined by copying remote "datacontrol.h" database to LOCATION in a local repo by adding all implemented UIDs to tracking list with logs in a reserved UIDs list.
- To reserve a UID, write the reserved UID information to the database via API command line. Each UID will be assigned next available UID slot, echoed to the command prompt.
- Reserved information is written in ascending numerical order into remote datacontrol.h as a comment.
- Reservation UIDs are automatically assigned, so no conflicts should occur.
- To convert a UID from reserved to implemented, command line call "- -implement <uid> ". The prompt will be asked to confirm implementation of details used to identify itself. If a UID is approved, the script will convert the change from a temporary comment to an actual remote database slot.
- In the data control process, a local non-database "datacontrol.h" is not effected until a successful change is push directly to data control database. Pushing additions to origin branch will fail if source code management (I.E. Git) changelog and database "datacontrol.h" are not coherent. To check if source code and database are coherent use the build validation flag. The call is automated before creating pull request on code.
- Backward compatibility means it conforms to coding standards checks with a UID data structure addition, and invalid changes throw an error. Builds must be enforced as a standard prior to requesting code review and a review requestor must a Promote Request validation script.

Compare your Local datacontrol.h to Remote

- A developer can call the compare flag anytime you want to safely compare your local copy of "datacontrol.h" to the remote "datacontrol.h", when developing changes to UID data structures. Any local changes will be overwritten to your local "datacontrol.h" with a temporary UID(s) (temp-uids). The command prompt API only prints out differences between local and remote to command line.

```
datacontrolgen.py --compare
====UIDs in local that don't match remote====
<uid>
====UIDs in remote that don't match local====
<uid>
0 error(s) found

- All differences and conflicts to local "datacontrol.h" must be resolved before
pushing/merging.
```

Periodic Maintenance of repo-meta

- Timestamp and user who pushed and approved a uid information is kept in repo-meta. Regular removal of UIDs not approved within the requested development timeline. Cleanup allows for incomplete change

or late features are reviewed. The default timeframe is 90 days so if developer changes require more time it must be approved and less than the timeline to product alpha mile stone.

```
$ python datacontrolGenMeta.py --delete <uid>
0 error(s) found
```

- The superuser who is approving the merge to trunk will not be able to if it has not been created, and will thus know it has been cleaned. UID changes corresponding to that pull's history should be undone and --implement called again.
- Similarly, if the uid is assigned before

Deprecating a Telemetry UID

Deprecating a UID is an instance of edit, where the struct product/builds of the latest major, minor is changed to empty.

Deprecate can be called on both reserved temp-UIDs and implemented UIDs. To deprecate, run:

```
$ python datacontrolgen.py --deprecate [<uid>,<telemetry-version>]
0 error(s) found
```

<telemetry-version> options are currently: 1.0 or 2.0

Deprecation affects only the latest major minor, but this is the "official" state of struct

Implementing a Telemetry UID

(Pending implementation) note: Attempting to implement a temp-UID that is not present in the source results in this error:

```
$ python datacontrolgen.py --implement
INFO: No change to remote datacontrol H repo detected. Note NOTHING HAS BEEN PUSHED TO REMOTE
0 error(s) found
```

(Pending implementation) : Calling implement could automatically change all temp-uid mentions in the source code to match the assigned UID. A uid check could be integrated into builds to ensure all UIDs are permanent to pass the build-time UID check.

Definitions for C/C++ Code

Data Type - A Data type is the definition of the data object and the elements based on an inherited object and/or standard C elements.

Instance - The following is the declaration of the data object.

Data Control Identifier - The data control identifier is the predefined detection template for the automatic generation of objects. These include the required version tracking information to ensure we can generate any object. The definitions of the pieces of information for each element is documented in the code through the 'datacontrol.h' file.

Guidelines

Data Control Template and Comments

Use the `///<` style since the following is the tag within a structure we use to check the data control components. For comments related to data object sub-elements, ensure the data item has a truly simple self contained. The self explanatory context is necessary for not only usage in firmware; for a point of context is telemetry is used by: validation, application, tools, and customers engineers.

Aliases

```

Version Number.
    thermalSensorWarning_e primaryThermalSensorWarning; // Indicates if selected sensor
exceeded threshold limits.
    uint32_t rsvd4[32]; // Extra padding to allow more dies
in "nandTemperature".
} thermalSensorV1_t;

thermalSensorV1_t thermalSensor;

```

Pros

- Improve readability by simplifying a long or complicated name.
- Communicate the operation and features clearly.
- Allows for auto parsing and detection of scripts so developers do not have to write these.

Cons

- Requires developers to learn the data structure features when usage is unclear decreasing velocity.

Summary

- Write comments to enable real detection of the auto parsing.
- Clearly understand the hierarchy to ensure any user can completely understand the functionality.

Annotations

When using these with the auto generation, these can get tricky to detect and should be used with caution.

```

/**
 * Temperature sensor parameters.
 */
typedef struct
{
    ///< Data Control Tracking information for Telemetry.
    ///< uniqueIdentifier = uid_ThermalSensor_e;
    ///< major = THERMAL_VERSION_MAJOR;
    ///< minor = THERMAL_VERSION_MINOR;
    ///< size = THERMAL_SENSOR_EXPECTED_SIZE;
    ///< duplication = oneDuplication;
    ///< dataArea = oneDataArea;
    ///< persistence = reconstructedPersistence;
    ///< dependency = single;
    uint8_t majorVersion; // Temperature Structure Major
Version Number
    uint8_t minorVersion; // Temperature Structure Minor
Version Number
    thermalSensorWarning_e primaryThermalSensorWarning; // Indicates if selected sensor
exceeded threshold limits
    uint32_t rsvd4[32]; // extra padding to allow more dies
in "nandTemperature"
} thermalSensorV1_t;

typedef struct thermalSensorV1_t thermalSensor_t;
#ifndef BAD_CODE
    typedef struct thermalSensorV1_t thermalSensor; // BAD! We cannot detect the version
information here
#else // GOOD_CODE
    thermalSensor_t thermalSensor; // Good this is a great way to manage several versions of the
same type!
#endif // BADCODE

```

Pros

- Aliases can improve readability by simplifying a long or complicated name.
- Aliases can reduce duplication by naming in one place a type used repeatedly in an API, which might make it easier to change the type later.

Cons:

- Aliases can create risk of name collisions
- Aliases can reduce readability by giving a familiar construct an unfamiliar name
- Type aliases can create an unclear API contract: it is unclear whether the alias is guaranteed to be identical to the type it aliases, to have the same API, or only to be usable in specified narrow ways

Summary:

- Do not use these unless you are generating the good code case.

Forward Declarations

Do not use these and instead use a #include header when necessary. The usages of forward declaration does not encourage modular code design and the auto detection tools cannot resolve these type of challenges since it requires multiple passes to get the information of the data structure.

Pros

- Forward declarations save compile time by limiting the files needing to be opened to create a symbol list. Not using these forces the compiler to completely recompile versus incremental changes.

Cons

- Forward declarations hide dependencies for header file changes. These hidden items make API owners not able to see visible changes across compilations to ensure the parameters require a new namespace.
- These hide the symbols for namespaces std:: and produces undefined behavior in the compiler. The fault is significant in the HAL/PS layer of the firmware code.
- It can be difficult to determine whether a forward declaration or a full #include is needed. Replacing an #include with a forward declaration can silently change the meaning of code:

Forward De-clair Usage

Usage example of a forward declair in C++.

```
// b.h:
struct B {};
struct D : B {};
// good_user.cc:
#include "b.h"
void f(B*);
void f(void*);
void test(D* x) { f(x); } // calls f(B*)
```

If the #include was replaced with forward decls for B and D, test() would call f(void*).

- Forward declaring multiple symbols from a header can be more verbose than simply #includeing the header.
- Structuring code to enable forward declarations (e.g. using pointer members instead of object members) can make the code slower and more complex.

Summary

- Avoid forward declarations of entities defined in another project.
- When using a function declared in a header file, always #include that header.
- When using a class template, prefer to #include its header file.

Data Inheritance

When a structure inherits from a base structure, it includes the definitions of all data the base defines.

Pros

- Forward declarations save compile time by limiting the files needing to be opened to create a symbol list. Not using these forces the compiler to completely recompile versus incremental changes.

Cons

- Implementation reduces the code size by using existing types.

- Add a recursive data tracking nature and dependency to objects. The following can cause a recursive relation between objects and if one object changes then we have to split the data object. For items, which need the same data type declare an array or use explicit types to reduce type recursion.

Summary

- All inheritance should be public. if you want to do private inheritance, you should be including the instance of the base class.
- Do not overuse implementation inheritance. Composition is often more appropriate.
- Multiple inheritance is permitted, but multiple implementation inheritance is strongly discouraged.

Template Guidelines

Information around Data Control Comments

The data control ecosystem is actually a cross-language translator solving the language semantic constraints of the source and destination languages. The initial implementation was constructed on LLVM and since then the telemetry working group decided to pursue a faster path of integrating the detection through the GHS front/back-end APIs. As a result the implementation is restricted to what we can access; thus, there are constraints for the code comments. For example, all of the code template blocks must be within a data type lines of code this means. The limitation of within the lines of code means the comment block has to be in the data type or in the case of tracking not possible on the data type the instance name must have the comment block on that line of code.

Type Example

```
#define TARGETEXAMPLE_MAX 16      ///< Data array blocks
#define TARGETEXAMPLE_RESERVED 8 //;< Total blocks reserved for minor version expansion.

typedef struct
{
    ///< Data Control Tracking information for Telemetry.
    ///< uniqueIdentifier = uid_targetExample_e;
    ///< major = TARGETEXAMPLE_VERSION_MAJOR;
    ///< minor = TARGETEXAMPLE_VERSION_MINOR;
    ///< size = TARGETEXAMPLE_EXPECTED_SIZE;
    ///< duplication = oneDuplication;
    ///< dataArea = oneDataArea;
    ///< persistence = reconstructedPersistence;
    uint32_t    idNumber;           ///< Identification number of the object
    uint16_t    majorVersion;       ///< Major version number of the object
    uint16_t    minorVersion;       ///< Minor version number of the object
    uint8_t     data[TARGETEXAMPLE_MAX];   ///< A counter value used to detect the usage of
a data block
    uint8_t     data[TARGETEXAMPLE_RESERVED]; //;< Reserved for future expansion and data
alignment
} targetExample_t; //;< Target example is a template of how we add tracking information.

#define TARGETEXAMPLE_VERSION_MAJOR 1
#define TARGETEXAMPLE_VERSION_MINOR 0
#define TARGETEXAMPLE_EXPECTED_SIZE sizeof(targetExample_t)
targetExample_t targetExample; //;< Example counter tracker for data objects.
```

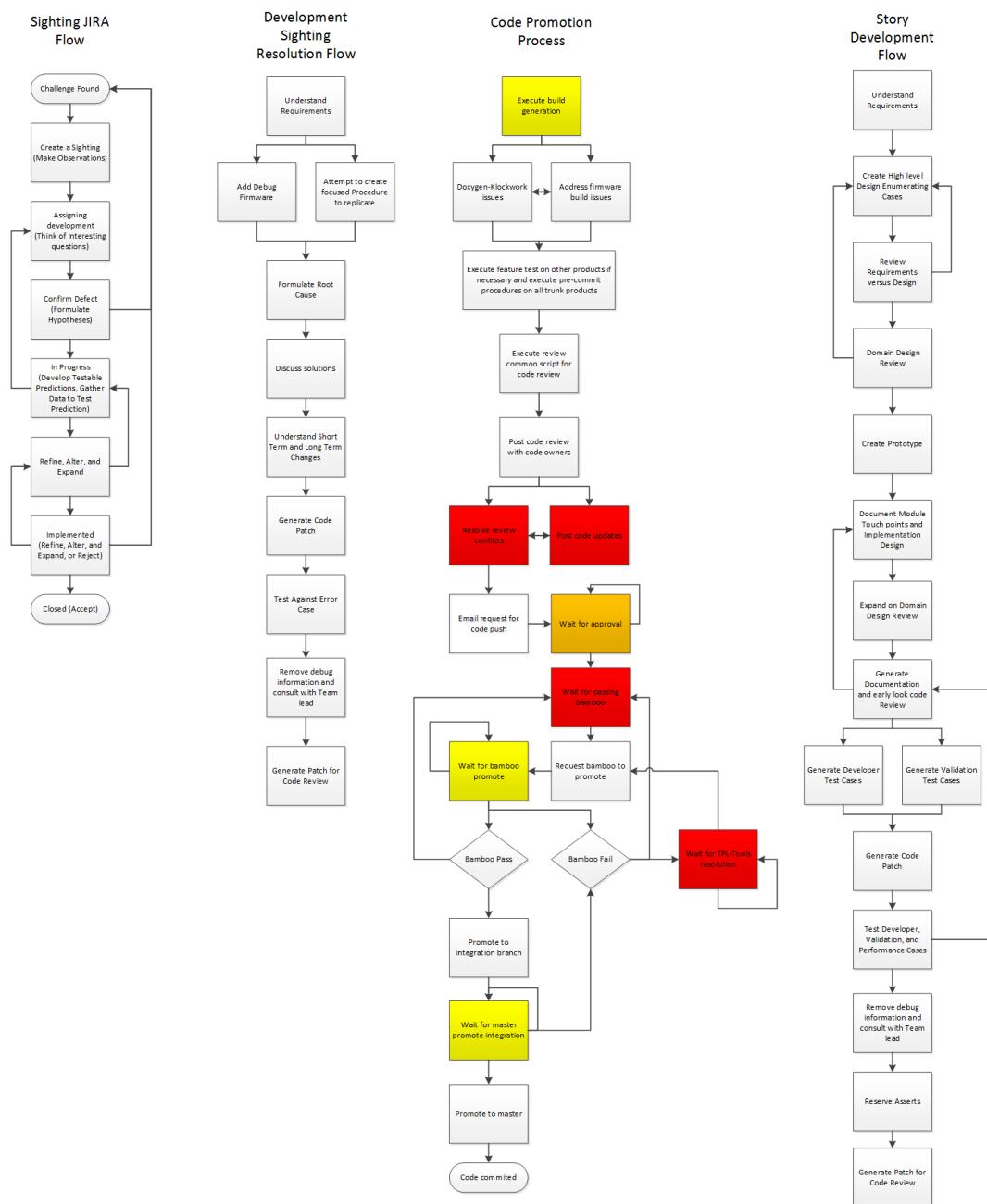
The example is a perfect instance of usage and detection for the compiler. We want to have a complete definition of all the object definitions. If added correctly auto generation can generate a parser by inputting the unique identifier value as one of the parameter conditions. Within the example you will notice the instance name is embedded in the identifier uid_targetExample_e. The exact instance name is a requirement for auto-detection since it used the firmware name to extract all of the information from the GHS symbol interpreter.

Instance Example Type Example

```
targetExample_t targetExample; ///< Data Control Tracking information for Telemetry. ///<
uniqueIdentifier = uid_ThermalSensor_e; ///< major = TARGETEXAMPLE_VERSION_MAJOR; ///< minor =
TARGETEXAMPLE_VERSION_MINOR; ///< size = TARGETEXAMPLE_EXPECTED_SIZE; ///< duplication =
oneDuplication; ///< dataArea = oneDataArea; ///< persistence = reconstructedPersistence; ///<
Example counter tracker for data objects.
```

The following is required to be all in one line of code because of the compiler constraints described above; therefore, the following is an exception for the auto detection flow.

Accelerating Code Velocity to Integration



Development, code, and promotion flow.

Actions Required

- Time per module
- Links to wiki documents.
- Invention Disclosure Form for automated mechanism
- Meta parsing scripts

Trunk Code Promotion Definitions

1. Execute build generation:

- The code build generation for trunk will build all targets for active products in development. These typically include development firmware, customer firmware, injection testing, and platform specific builds for each product line. The time to execute the build is done with the following command python buildgen3.py -makeasserttables -keepgoing is approximately 45 minutes to 60 minutes.

2. Doxygen-Klockwork issues:

- In generating the code, we run two scripts to do documentation analysis for code and static analysis to address conditions possibly resulting in a customer failure. These changes are necessary to maintain code maintainability and to address challenges in smaller granularity compared to large efforts.

3. Address firmware build issues:

- To ensure code is maintainable, we run the full builds across all products to test product based build switches or product specific features. Due to disjoint features among the products, our compiler will either add functionality in the libraries as necessary. Resolving build challenges typically attempting to catch the library paths not exercised in the story for the current development.

4. Execute feature test on other products, if necessary, and execute pre-commit procedures on all trunk products:

- Depending on the product family, a feature may exist in other products and if there are disjoint versions the feature test needs to be executed to ensure compatibility among deployments. Pre-commit procedures are necessary to catch gross failures that stop product development in our common code model.

5. Execute review common script for code review:

- The expectation is for engineers not knowing the domain ownership, the names generated will point them to the correct stake holders. These are expected to be concrete ownership.

6. Post code review with code owners:

- The code is typically posted to our fisheye-crucible online tool at which stake holders can mark items as conflicts and make suggestions to change. The challenge here is there is no clear decision maker for a product or common code model.

7. Resolve review conflicts:

- Review conflict can be marked as code style, standard, etc. marked with criticality accordingly. The update for these changes should focus the engineer's scope. For example, the architecture should check if it maintains the encompassing customer needs and domain owners can note for complications or extensibility.
 - The changes here can range from code style to encompassing requests. The limits of the code review scope are often diverged into tasks that are related to technical debt. For changes outside of the scope of the active development, it is proposed the stakeholder defines the development such that the story will follow the development flow for review. The timeline for the code review is uncertain most of the time as responsiveness is lacking from owners and can be difficult to contact. A rule for resolution needs to be in place such as a bot to remind engineers and when no action is needed managers are automatically notified to do follow up with the engineer owner. In

the event time expires past the deadline, the code should be allowed since the developer is blocking progress.

8. Post code updates:

- The updates are to address conflicts and overlying functionality concerns.
 - The challenge for this is as the code progress from version to version is without explicit expectations and goals set the update procedure can diverge into an infinite update loop.

9. Email request for code push:

- The request for a code push is to communicate the JIRA, code review, summary of the challenge, complexity, scope, and testing procedures.
 - The challenge is information is often communicated in the JIRA or code review and the procedure assumes the developer was not diligent.
 - The email format varies significantly across programs and long branches, and requires non-negligible manual effort to populate needed tables and provide relevant information. An outcome of this is that developers populate the bare minimum in the required fields, which can result in an approval email that contains little to no information of use to a TPL approving the request, which either results in pushback or a rubber stamp, neither of which improve velocity.

10. Wait for Approval:

- Approval is mechanism to ensure pushes are such that changes do no collide together.
- The challenge here is the approval should have been known through the trunk synchronization mechanism and the additional time to review slows the code promotion process a day to two.

11. Request bamboo to promote:

- The process to request is for when the code has been integrated to the closest integration working node. The code is executed on our tools to see if there are common path functionality issues.

12. Wait for bamboo promote

- Waiting for bamboo to promote is a mostly polling process.
 - The challenge here is the promote process should be interrupt driven so the developer is notified of the status and can work without having to check the status manually.

13. Wait for TPL-Tools resolution:

- These are challenges the encompassing tools have difficulty in coverage. These can include code changes, tools update, server up/down time, and infrastructure.
 - The challenges here are there are several avenues of failure and each subsystem should appropriate secondary systems to restore the functioning state.

14. Promote to Integration Branch:

- When bamboo has pass the code is then promoted to integration branch such that code is now a candidate to master.

15. Wait for Master Promote Integration

- The code goes through a secondary flow that check the integration merge with the master node as a secondary mechanism to attempt to catch intermittent challenges.
 - The challenge for the promotion of integration to master is the actual landing of code in the main repo for continuous validation is unknown and the status is not apparent in our JIRA tools. Tracking these changes is a manual process and requires searching for the individual change set.

16. Promote to Master

- The promotion to master is after the completion and the code is now in transition to the master code branch.

17. Code Committed

- The code at this stage is known to be in the master branch and ready for continuous validation.
 - A mechanism linked to JIRA or a notification to the stakeholders would be useful for blocking items.

JIRA Sighting Flow

1. Create a Sighting (Make Observations)

- Define the problem by creating a sighting based on Triage information which is a formulation of a question based on observation from triage members. The triage process will collect evidence, experiments, and correlation between known issues.
- Common information will include detailed information on system configuration (driver, firmware, tools version), time to failure, default failure analysis data, steps to reproduction, and last known good configuration.
- A sighting created moves from new to assigned development at this point.
- Jira Procedure
 - Mark Issue Type as Sighting.
 - Update JIRA Main: Summary, Priority, Exposure, Program, Affected Products, Suspected Problem Area, Affected Products, Submitter Org, Assign Team.
 - Update JIRA Details: Development Platform.

2. Assigning development (Think of Interesting questions)

- Depending on the workload of the team the program manager, system technical product lead, and technical leads will review the debug data.
 - Upon first pass of the data, each member will quickly review the debug data and determine the focus team appropriate for further investigation.

- When strong correlation is determined between known issue and the concerning area at hand the development by determining a highly similar failing case signature, then the development will be marked as duplicate and linked as 'is duplicated by' following the closure of the parent node the sighting will be re-evaluated such that the failing case is satisfied otherwise the issue will be moved to confirm defect and the link will be moved to 'is related to'.

3. Confirm Defect (Formulate Hypotheses)

- Technical team lead and team member will move sighting to confirm defect based on the experiment, analysis of data, and reproduction.
- The team members will specify the requirements and identify variables violated in the defective case.
- If there are multiple failure conditions then additional JIRAs will be generated for each challenge and mark as related to.
- Update JIRA Main: Source of Error.

4. In Progress (Develop Testable Predictions, Gather Data to Test Prediction)

- Based on evidence, the team member will investigate the fault area beginning to formulate a predictive mechanism or existence of defect in design.
 - The process will involve understanding underlying mechanisms and system interactions to classify the failing case. The development of the failing case will involve understanding the control flow and data flow graph with the behavioral expectation to the error classification.
 - The general failure condition will show a trace of valid conditions until one is not met in the flow such that the coverage of all cases are depicted in a manner such as Boolean algebra, truth tables, sequence diagrams, finite state machine (deterministic finite automata), etc.

- The developer will develop and clarify the focused experiment, cases, and predictions for the root cause. The developer will gather data to conclusively replicate the failure or conditions that lead to such a situation.
 - The developer will cultivate and expand a hypothesis on the finding to give exact clarity and repeat the development until the conclusive root cause is understood. When the developer exhausts resources or strategies the failure will be promoted to a brain storming session or working group to further understand failure to develop a collective hypothesis or explore mechanisms until a relevant formulation is reached.
- Refine, Alter, and Expand
 - Upon proving the root cause of the failure, the developer will understand and propose solutions based on the architecture and implementation to address the concern and maintain future extensibility.
 - For special cases, a workaround or short term fix until a long term solution is provided.
 - Build a prototype to test defective case proving solution definitively resolved the original defect.
 - Before completing the solution, the developer must ensure the requirements are met and communicate results to technical lead, system technical product lead.
 - The solution will follow the standards defined for the product library and appropriate reviews before promoting to live library testing.

5. Implemented (Refine, Alter, and Expand, or Reject)

- After the developer has completed failure case resolution the system technical product lead will communicate the attention of the resolution case where the solution will be tested in an independent controlled environment the failure was produced

in confirming the solution is appropriate in the common product library.

- If a specific hardware workaround is necessary, a development story will be created and reviewed in the cross functional domain forum. In the case, the change is between hardware revisions and a hardware change is expected to commit a change. Then an additional story is to be created for the development platform to be tested against and the final platform. All of these stories will be linked as found by the original sighting.

6. Closed (Accept)

- Duplicate or linked issues are to be reviewed and verified against the solutions and if the change address the failure the duplicate will be closed as a duplicate, otherwise, the JIRA will be unmapped and reopened to assigned development.
- Solution is determined to be adequate for test case and corresponding horizontal technical leads should be made aware of the item resolution.
- Follows to the solution that alter the architecture or expose failure conditions must be communicated to forward looking teams in a functional change in the ASIC, Firmware, Media, and/or Tools.
- Update JIRA:
 - Fixed in Component, What Changed, Resolution Description, Affected Platforms, Milestone.

Detect Data Leaks (Example)

Security leak detection automation is an automated process that aims to detect security leaks in telemetry binaries.

The goal of the automated security leak detection tests is to ensure that

1. No critical security parameter (CSP) shall be part of telemetry logs in any form, including encrypted
2. No CSP shall leave the production device in any form, including encrypted
3. Security leaks in telemetry logs, if any are detected, are addressed in a timely manner
4. The manual security leak analysis effort is minimized
5. For a list of sensitive data, please refer to Data Control classification.

To automate the security leak detection tests only detect leaks from the run-time generated security keys and Random Number Generator (RNG) data, which is the most difficult part to automate and is made possible for automation with instrumentation.

There are other sensitive data in sensitive data that we don't have security leak tests for yet, such as user data, ID, Opal password, etc. Sensitive data can be set to known in the test, it is easy to develop tests for them without the need of instrumentation.

Instruments the key generation, DRBG (random data generation), key wrapping, and key unwrapping operations to force a known pattern instead of a random number. We can then search for this known pattern in the telemetry logs to identify any security leaks. The pattern can represent any potential customer data as well.

```
import re, ctypes, binascii

class MetaDataStructure(Structure):
    _fields_ = [ ("metaData", ctypes.c_byte()*sizeof(Structure)]
```

```

def searchSecurityLeak(metaData, pattern = '\xE7', repeatCount
= 32)
"""
Brief:
    searchSecurityLeak() - Scan the telemetry log for security
leaks

Description:
    In order to use this method to search for security leaks in
telemetry log, security pattern injection has to be enabled on
side first. Once that's done, we can search the pre-defined
fixed pattern in the telemetry log to identify if there's any
security leaks in the log.

Return Value(s):
    True if security leaks are detected in the telemetry log
    False if no security leaks are detected in the telemetry
log
"""
# The pattern below needs to match the pattern on side when
# security pattern injection is enabled on side
SECURITY_PATTERN = pattern
# Minimum length in bytes of security keys or random data
# generated on side
REPEATED_BYTES = repeatCount
data = bytearray(metaData)

# I.E regular expression '(\xE7){32}' representing 32 bytes of
# 0xE7
# search 32 continuous bytes of E7 in the telemetry log for
# security leaks
regPattern = '({}){{{}}}'.format(SECURITY_PATTERN,
REPEATED_BYTES)
regex = re.compile(regPattern)
result = False
# Find all non-overlapping matched instances in the log and
# print out the byte offset in the log of the matched instances
# Which are the security leaks
for matchedInstance in regex.finditer(data):
    result = True
    byteOffset = matchedInstance.start()
    print("Found security leak pattern ({} bytes of {}) at byte"

```

```
offset {}".format(REPEATED_BYTES, '0x' +
binascii.b2a_hex(SECURITY_PATTERN), hex(byteOffset)))
return result
```

Mentoring

What is a Mentor?

- An individual who takes an active interest in helping a protégé set and achieve goals in an academic, industrial, or other environment.

Mentor Behaviors in Successful Guidance

- Act as a source of information on the culture, norms, and expected behaviors
- Tutor specific skills, provide effective strategies
- Give feedback and provide coaching
- Serve as a confidante in personal crises and problems
- Demonstrate confidence in protégé's ability
- Assist in planning a career path
- Empower the protégé make own decisions
- Maintain integrity of the relationship between the protégé and the natural supervisor
- Listen >> talk
- Brainstorm solutions
- Communicate high expectations
- Recovers from setbacks
- Knows when to seek advice

Protégés Traits

- Interested in Receiving Advice
- Receptive to Constructive Criticism
- Spend time preparing for mentoring session
- Unafraid of asking probing questions

Characteristics of a protégé needed to be successful

- Learning style
- Personality style

- Modality preference
- Cerebral Hemisphericity
- Career interests
- Expectations of the mentoring experience
- Cultural background

Creating a path

- Use scientific method to explore the unknown
- Professional strategies
- Personal strategies
- Career search strategies
- Confidence boosting strategies
- Independence and autonomous strategies

Continuous Development and Expansion

- Active learning is more lasting than passive learning
- Thinking about thinking is important (Meta-cognition)
- The level at which learning occurs is important
- Bloom's Taxonomy (In order of depth to breath)
- Evaluation: Making decisions and supporting views; requires understanding of values.
- Synthesis: Combining information to form a unique product; requires creativity and originality.
- Analysis: Combining information to form a unique product; requires creativity and originality.
- Application: Combining information to form a unique product; requires creativity and originality.
- Comprehension: Restating in your own words; paraphrasing, summarizing, translating.
- Knowledge: Memorizing verbatim information. Being able to remember, but not necessarily fully understanding the material.
- Time Management
- Connections
- Check in evaluation of learning progression

Topics Ideas for Discussion

- Introduce personal background.
- Determine expectations.

- Explain technical politics.
- Successfully expanding networking.
- Determine what successful means at Intel.
- How to balance workloads and determine commitment time frame.
- How to establish credibility.
- Team leadership techniques.
- Feasible technical paths at Intel Corporation.
- Develop growth plan.
- Path to promotion.
- Research projects at Intel and how to handle NDAs.
- Patient process, publications, white papers
- Appropriate process for driving a new ideas.
- Managing technical meetings and behave according to excellent expectations.
- Organization Contribution IEEE, ACM, SHEP
- Conferences
- University Interactions
- Industry Influence
- Technical Advisory Spokesperson
- Aligned Strategy

Building of Personal Brand

Ann Bastianelli explains how to use your skills and build a personal brand through being self-awareness, telling your story with you life, and deepen your relationships. - People do not care about what you know, until they know how much you care. - A powerful personal brand means you will: lead more, win more, and earn more - Surround yourself with those focused on a goal and not on you. - When given feedback, treat it as gift and become self aware to the provided information.

@todo <https://www.youtube.com/watch?v=hcr3MshYe3g>

How to create an Impactful Career Development Plan Materials

- Values Cards: <http://thegoodproject.org/toolkits-curricula/the-goodwork-toolkit/value-sort-activity> & <http://www.uhi.org/wp-content/uploads/2013/08/FINAL-Value-Card-Set-082313-CMS.pdf>
are 2 of MANY options

- Novation's Stages of Contribution:
<https://www.kornferry.com/institute/the-four-stages-of-contribution>
- 9 Box HR Succession: <https://www.pageuppeople.com/en-us/2017/06/16/make-9-box-succession-planning-work-suc001/>

External Material Creditors

- Saundra Y. McGuire, Ph.D.
 - Director, Center for Academic Success
 - Adj. Professor, Department of Chemistry
 - Louisiana State University

External Reference Material

- Murray, M. and Owen, M. (1991). Beyond the Myths of Mentoring. San Francisco, CA: JoseyBass.
- Peddy, S. (2001). The Art of Mentoring: Lead, Follow, and Get Out of the Way. Houston, TX: Bullion Books.
- Taylor, S. (1999). Better learning through better thinking: Developing students' metacognitive abilities. *Journal of College Reading and Learning*, 30(1), 34ff. Retrieved November 9, 2002, from Expanded Academic Index ASAP.
- <http://academic.pg.cc.md.us/~wpeirce/MCCCTR/metacognition.htm>
- Zull, James (2004). The Art of Changing the Brain. Sterling, VA: Stylus Publishing.

Guide to Action Required (ARs)

Before assigning or receiving ARs; review the following:

Do(s) for Communication

- Effort: The effort should be reasonable given the variables of the task.
- Specific: ensure there are details of what exactly to be accomplished
- When: Confirm the date and ensure why the target is critical.
- Who: Be clear there is an owner for the AR

- Confirmation: To give an action the receiver must and agrees.
- Owner Agree(s): Ensure the owner is present, agrees, and knows the details. The owner should be sure the AR can be complete or offer details for actions if it cannot be.
- Priority: Any assigned AR should be confirmed with its priority and can be met with other deliverables on the owner.
- Ask Question(s)
 - Clarification: The owner should ensure they understand the key aspects of an action item and if they are not clear do not accept and follow up.
 - Division: If the action is similar between two individual ensure each deliverable is disjoint or has a collaborative sync to ensure everyone is on the same page.
 - Fit: Ensure the action is a correct fit with your skill set or role.
 - Follow up: Before the due date of the AR; it is good behavior to give updates on the trend of completion. The trend is valuable for additional help and managing inbound change.
 - Digital: To ensure expectations are set; follow up each AR with an Email or a set of meeting notes for the deliver to agrees with.
 - Decision: For the AR conclusion, will you have a decision on the overall drive of the effort?

Avoid for Efficiency

- Ambiguous: Actions in which do not change decisions or execution.
 - Items where a decision is blocked pending gathering more data. These actions require a separate confirmation and should be reviewed to ensure the tasks are not nebulous.
- Duplication: Similar ARs to multiple people; each owner should have a separate AR. In the event, it is a team AR ensure there is a single owner for delivery or team.

Research and Development Guide

The content is designed to be a guide for successful mentoring and guidance in the technical ladder. Within the internship, we will cover mentoring, taking actions, and the guide for becoming a successful innovator. The guide is maintained by Colorado University Professor and Researcher Joseph (Joe) Tarango. Joe has successfully mentored engineers at organizations such as: National Science Foundation, Google, Intel, Facebook, and international startups.

Internal Career Connections

- <https://careerconnections.intel.com/connector-profile/?entry=7186>

Linkedin

- <https://www.linkedin.com/in/joseph-tarango-451695a2/>

Overview

Intel has a well-developed internship program serving thousands of students who are preparing for real careers outside of college. Our goal is to provide real-world, thought-provoking internships to undergraduate and graduate-level students. Historically, more than 60 percent of our interns go on to accept full-time opportunities with Intel after graduation. There is no better way to learn than to experience things first-hand.

Intel internships offer students a chance to get real-world experience with ownership of projects from day one, as well as the opportunity to develop a network of contacts for their future. Intel managers encourage students to take the initiative and develop programs that meet their particular interests.

Reference <https://jobs.intel.com/page/show/internships>

As an intern at Intel, you will enjoy a variety of benefits including:

- Real-world experience with leading-edge technologies
- Competitive salaries
- Networking with Intel managers and executives
- Access to Intel University classes for professional and personal development

- Consideration for full-time employment when you graduate

An Inside Look:

What's It Like to Intern for Intel?

- Eligible candidates for temporary full-time positions are students who have not yet graduated and are working towards a relevant Bachelor's, Specialist's, Master's, or PhD degree from an accredited academic institute.
- Before applying, please review our internship requirements and inquire with your school about receiving credit for your internship. Internship opportunities may not be available at all locations. Check the job listing for additional information.
- Submitting an application does not guarantee an intern assignment, and an intern assignment does not guarantee regular full-time employment upon completion of your degree program. An Intel interview panel reviews all applicants and selects all interns. Applicants are screened for communication, interpersonal, and job-related technical skills.

Introductions

The first assignment is to complete the enumerated items below. I posted examples below each.

1. Post a picture or a link to your picture.
 - <http://www.cs.ucr.edu/~jtarango/> (Links to an external site.)
2. Name
 - Joseph David Tarango
3. Where did you grow up?
 - California, USA
4. Major and Focus
 - I.E. Computer Science and Engineering, Computer Architecture

5. Programming Languages you are proficient at.

- I.E. C++, VHDL, Python

6. What is your Interest in Advanced Computer Architecture or Domain Area?

(https://en.wikipedia.org/wiki/Comparison_of_instruction_set_architectures)

- Computer architecture has always been an interesting topic. It combines art and science to design system to meet real time needs. I have used computer architecture to design many ISA based systems including:

- MIPS
- OpenRISC
- SPARC
- x86
- RISC-V
- ARM/A32
- Thumb/T32
- A64

- I want to create the lowest power real-time EKG smart processor to improve the human condition and save lives.

7. What is your expected Objective?

- To teach such that by the end of the course all of the students are ready to do architecture in academia and industry.

8. What do you want to achieve in your career? (shoot for the moon and if you miss you will still be among the stars)

- Industrial
 - Drive innovation and achieve Intel Senior Fellow status
 - <https://newsroom.intel.com/biographies/senior-fellows>
 - <https://newsroom.intel.com/biographies/fellows>
- Academic

- Achieve the Turing award
 - https://en.wikipedia.org/wiki/Turing_Award
 - <https://amturing.acm.org>

9. What would you like to focus on? (Choose one below and explain why)

- Patent Process
 - Patent Application with Technical White Paper
 - Model, Simulation, Emulation, or Prototype
 - Github repository with source code using APL 2.0
 - Presentation
- Reproduction of Research in Computer Architecture
 - Conference style paper with Literature survey and technical results from your instantiation.
 - Github repository with source code using APL 2.0, if one exists for the project you must cite it.
 - Conference Style Technical Results
 - Model, Simulation, Emulation, or Prototype
 - Presentation
- Novel Research in Computer Architecture
 - Conference Style Technical Paper
 - Model, Simulation, Emulation, or Prototype
 - Github repository with source code using APL 2.0
 - Presentation
- RISC-V Open Source Contribution
 - Proposal
 - Github repository with source code using APL 2.0
 - Baseline
 - Modular Contribution
 - Technical documentation
 - Presentation

- Unique Project
 - Post details and minimum deliverables include:
 - Choose one either:
 - Conference style white paper
 - Detailed Technical documentation
 - Model, Simulation, Emulation, or Prototype
 - Github repository with source code using APL 2.0
 - Presentation

10. Write a Biography

- Joseph's undergraduate/graduate career at the University of California, Riverside (UCR) has allowed him to participate in several domestic and international projects. A few of significant projects have included: similarity search, generalized interfaces for hardware accelerators, hardware accelerator optimization, memory abstraction/standardization of hardware systems, and enhancements of Reduced Instruction Set Computing (RISC) processors. In these projects, he excelled in project management, coordination, and collaboration with his peers. Throughout these experiences, Joseph demonstrated determination, leadership, technical merit, and ability, to independently learn/improve skill sets. Collaboration with multidisciplinary groups taught him effective communication and problem-solving skills. Joseph has collaborated with research/technical groups such as Jacquard Computing, Pico Computing, Intel Corporation, Ecole Polytechnique Fédérale de Lausanne (EPFL), National University of Singapore (NUS), and University of Bern.

11. Share what you are passionate about science.

- My passion for science was ignited at an early age in the form of curiosity, and it all began at my grandparent's house. Every summer, I would visit my grandparents for

a month or so and each time I entered the garage I was enchanted by my grandfather's radio communication system. Its buttons, knobs, lights and sounds were too stimulating for my curious nature to resist. One afternoon, I decided to investigate my grandfather's elaborate radio communication system by decomposing it on the garage floor. Upon discovery, my grandfather was more impressed than angry because I was able to separate the parts by levels of similarity. Instead of punishing me, my grandfather decided he would teach me how to reconstruct the radio communication system. After a week of rebuilding, my grandfather not only taught me how to build a transistor radio, but also about the fundamental concepts of electricity, circuits, and radio communication. From then on, my grandfather encouraged my curiosity with a new project every summer and thus began my mechanically inclined nature.

12. Discussion

- Share a share a unique fact.
 - I have a soft spot for cats and I adopted mine after a friend could not keep him. Coincidentally, she named him Joseph. He is a Savannah and Domestic Mix T5 at about 17+ years old and he is so large I take him for walks on a leash around my condo complex.
- Share a share a trait.
 - A majority of my friends say I am one of the reliable and hard working people they have met.
- Share an achievement.
 - I achieved athlete of the year my senior year in High school for competing and placing in the highest levels of competition in: Olympic Wrestling (Folk/Freestyle), Football (Full/Tail back, outside-line backer, and special teams

receiving) Track & Field (100, 200, 400 meter dash), and Tennis (singles)

- Share something what you are thankful for.
 - I am thankful for the mentors I have had in my life including: my grandfather, high school mathematics teacher, academic advisor, and many more. Without them and embracing my potential; I would most likely would not have as rich of a life as I have had up to now.

Proposal

- How to write your first paper:
<https://ieeexplore.ieee.org/document/6526784>
- ACM template
 - <https://www.acm.org/publications/proceedings-template>
 - https://www.acm.org/binaries/content/assets/publications/word_style/interim-template-style/interim-layout-.docx
- Provide 3-5 pages in ACM format with the project proposal with 100 points total.
 - ACM Template Usage
 - Abstract
 - High level overview of hypothesis
 - Introduction
 - Describe historical context required for comprehension
 - Literature review of dependencies.
 - Hypothesis
 - Provide motivation for the challenge to be solved.
 - Diagram of high level context
 - Benchmarking Strategy
 - Ecosystem to construct the project
 - Methodology to compare against baseline
 - Deliverable

- Project simulation, emulation, model, etc.
- Verification Strategy
 - Methodology to ensure success
 - Unit Testing
 - Integration Testing
 - End to End Testing
- Research Detailed Plan
 - Timeline (by week)
 - Deliverable by week
- Conclusion
 - Expected results
- References (10 pts)
 - Literature review references
 - Project Github Creation
 - The assignment is to create a baseline repository for all of the development related to the course project. Ensure the repository is private and not visible to any one except for mentor. If somehow there is a divergence with an explanation why.

Instructions

- <https://product.hubspot.com/blog/git-and-github-tutorial-for-beginners>

Example Githubs

- <https://github.com/intel>

Adding Collaborators

- <https://help.github.com/en/articles/inviting-collaborators-to-a-personal-repository>

Github Visibility

- <https://help.github.com/en/articles/setting-repository-visibility>

Steps

1. Create a Github account
2. Ensure the repo is a private github repository 20 pts)

3. Add APL 2.0 license (20 pts)

- If you are using my code copy please add the following:
- Copyright and patent pending by Joseph Tarango (joseph.d.tarango@gmail.com). Do not use or redistribute without explicit permission.

4. Create the directory and file tree (40 pts):

- README.md
- LICENSE
- projects (folder): contains property based project files such a visual studio:, greenhills software, altera, etc.
 - makefiles
 - visual_studio
 - greenhills_software
 - altera
- src (Folder)
 - hardware
 - README
 - <Module Names> I.E. floatingPoint
 - <src>.v,vhd>
 - software
 - README
 - <Module Name> I.E. dynamicPointLibrary
 - makefile
 - <src>.h,hpp,c,cpp>
- documentation (Folder)
 - proposal (folder)
 - proposal_v<version number I.E. 1>_<name I.E. JosephTarango><creationDate I.E. 9-28-2019_12-01pmMST>.docx
 - figures (folder)
 - data (folder)
 - rawData (folder)

5. Add Collaborators

Project Checkpoint(s) at Week Cadence

- Based on your timeline each should have a deliverable; please submit what you expect. These cadence reports will carry 100 pts. In the event a week has a reduction in work days by 2 or more (due to holidays or vacation) then the week will be aggregated into the next report.

Provide a summary of the following

- What has been completed?
- What challenges have occurred?
- Are you on schedule? If not how will you get back on track.
- Do you need help? If so, be specific.
- How much time have you put into the project? Please put down the date, amount of time, and summary of Git pushes.
- On time delivery?

Project Video

Recorded video presentation. The video should consist of each section taking as long as it takes to be clear and concise. Videos of excess of 2 hours should be reviewed by mentors before publishing.

- Code review of project
- Walk through of all items in git repository
- Video instruction of how to execute code to replicate results
- Short trailer of patent/research work
- Media format rules

<https://support.google.com/youtube/answer/1722171?hl=en>

Recommended Reference Material

- https://www.ted.com/talks/nancy_duarte_the_secret_structure_of_great_talks?referrer=playlist-how_to_make_a_great_presentation
- <https://www.youtube.com/watch?reload=9&v=Unzc731iCUY&feature=youtu.be>

Example and Informational Demo

- http://open-zfs.org/wiki/Documentation/Read_Write_Lecture
- OpenZFS novel algorithms: snapshots, space allocation, RAID-Z - Matt Ahrens

ACM template

- <https://www.acm.org/publications/proceedings-template>
- https://www.acm.org/binaries/content/assets/publications/word_style/interim-template-style/interim-layout-.docx

Example of successful papers:

- http://www.cs.ucr.edu/faculty/philip/publications/pubs_by_years.html

How to write your first paper:

- <https://ieeexplore.ieee.org/document/6526784>

Paper: Provide 8 to 12 pages in ACM format with the project proposal.

- Abstract
 - High level overview of hypothesis
- Introduction
 - Provide motivation for the challenge to be solved.
 - Diagram of high level context
- Background and Related Work
 - Describe historical context required for comprehension
 - Literature review of dependencies.
- Technical Construction
 - Analysis of Challenge
 - Exploration Methods
 - Analytical Theory and/or Proofs
 - Architecture choices and cost analysis
 - Software/Hardware Detail Architecture
 - Design Diagrams, Flow, Construction
- Technical Solution Space
 - Algorithms used in the Architecture and Instance
 - Semantic and Implementation Details
 - Environment and replication information
- Experimental Evaluation
 - Bench marking
 - Ecosystem to construct the project
 - Methodology to compare against baseline
- Conclusion

- Future Work
- Acknowledgement
- References
 - Include github link and all cited references.

Final Project Presentation and Grading

- Provide the presentation slides for each project. The slides should target 30 minutes with 5 minutes for Q&A.
- Examples
 - https://www.ted.com/talks/nancy_duarte_the_secret_structure_of_great_talks?referrer=playlist-how_to_make_a_great_presentation (Links to an external site.)
 - https://www.ted.com/talks/chris_anderson_teds_secret_to_great_public_speaking?referrer=playlist-how_to_make_a_great_presentation&language=en (Links to an external site.)

Patent Construction

Types

- Utility: these are processes, manufacturing, and compositions of matter these typically do something in the industrial or technology process. In the context of matter, these would be a process for creating new compounds.
- Design: These are the functional or structural feature; which require a utility patent as well. The structural or appearance of the object requires the utility since by itself it would be obvious to patent.
- Plant: Distinctive features of a plant in which do not appear without cultivation. An additional component is the grafting or cutting of the plant in the seed to the mature state.

Features

The features of a quality patent are first and foremost the baseline of the document. The overall items should have clarity and conciseness on conveying these elements below.

- Novelty: The originality of the invention is the foremost piece to highlight. Patents without novelty; even with high quality will be rejected. The goal of a patent is to protect intellectual property of something new.
- Value-to-Intel: These include quantification of benefits in example
- Detectability: The techniques in order to show replication, use of a patent through the process, and design through observation.
- Implementability: The how to realize the technology with today's state. For example in code, we may use a programming language to create software or hardware. In the physical world this would be the process in how to change in aspects of the material through technologies such lithography.

General Guidelines

1. Define acronyms on first usage.
 - I.E. PRD is product requirement document
2. Diagrams for aiding in the invention should be present in the first two pages. These diagrams need to be clearly labeled, explained, and referenced in the writing.
 - Each of the stages of the patent
3. Use precise language such that it can be clearly seen it is better. Do not use optimal or similar wording unless it can be proven.
4. Title: Typically patents include method and apparatus with these in mind:
 - Distinction between a claim to a product, device, or apparatus, all of which are tangible items, and a claim to a process, which consists of a series of acts or steps.
 - Apparatus claims cover what a device is, not what a device does.

- A process, however, is a different kind of invention. It consists of acts or steps, rather than tangible things. A process, therefore, has to be carried out or performed.
- Sale of an apparatus capable of performing the patented method is not a sale of the method. A method claim is directly infringed only by the entity usurping the patented method

5. Construction

- Problem definition provide a definition for the class of challenges not explored or apparent in usage. These cannot include properties of nature or strict mathematics.
- Previously solutions state what is the current known methods; in which, the patentee writes is not to search or explore before writing the patent.
 - Disadvantages within the space are used to point out the difficulties or gaps in the known methods.
- Short summary is the overall idea pointing out the features with any theoretical or empirical value.
 - Advantages are points in which the idea makes the challenge problem space more explored address the problem directly.

6. Detection Considerations

- Reverse engineering, 0.5 weight not practical
- Hardware/Software Telemetry.
- Binary instrumentation.
- Data injection for behavior trace paths according to the stages.
- Compiler level through pin with static or dynamic compilation.
- Latency, throughput, and the Intermediate Representation (IR) similarity core is high.
- Deterministic Finite Automata (DFA) or Finite State Machine (FSM) creation from sequences of commands, inputs, outputs, and actions.

7. Invention Details

- These are the bulk of the patent application and for the most part could be considered the appendix for the attorney to use in filing the formal application. This section not be required to get the high level idea; however, it is crucial to providing the methods and apparatuses.

Python Module Index

S

S

src

[src.software.autoAI.mediaPredictionRNN](#)
[src.software.autoAI.nlogPrediction](#)
[src.software.autoAI.NNStateBasedProcessing](#)
[src.software.autoAI.transformCSV](#)
[src.software.autoModuleAPI](#)
[src.software.axon.axonInterface](#)
[src.software.axon.axonMeta](#)
[src.software.axon.axonProfile](#)
[src.software.axon.packageInterface](#)
[src.software.container.basicTypes](#)
[src.software.container.indirection](#)
[src.software.dAMP.gatherMeta](#)
[src.software.dAMP.iSOM](#)
[src.software.dAMP.reportGenerator](#)
[src.software.datacontrol.dataControlGenMain](#)
[src.software.DP.preprocessingAPI](#)
[src.software.estimationROI](#)
[src.software.guiCommon](#)
[src.software.guiDeveloper](#)
[src.software.guiLayouts](#)
[src.software.guiOneShot](#)
[src.software.guiTests](#)
[src.software.JIRA.analysisGuide](#)
[src.software.JIRA.classCommonReports](#)
[src.software.JIRA.classGetJiraIssues](#)

[src.software.JIRA.classHtmlGen](#)
[src.software.JIRA.classJiraIssues](#)
[src.software.JIRA.classJiraIssuesList](#)
[src.software.JIRA.classSupportFiles](#)
[src.software.JIRA.executeJiraMining](#)
[src.software.JIRA.jiraEmbedding](#)
[src.software.JIRA.jiraSimAnalysis](#)
[src.software.MEP.loadAndProbeDrive](#)
[src.software.MEP.mediaErrorGUI](#)
[src.software.MEP.mediaErrorPredictor](#)
[src.software.mp.order](#)
[src.software.parse.autoObjects](#)
[src.software.parse.bufdata](#)
[src.software.parse.headerTelemetry](#)
[src.software.parse.intelObjectIdList](#)
[src.software.parse.intelTelemetryDataObject](#)
[src.software.parse.intelVUTelemetry](#)
[src.software.parse.intelVUTelemetryReason](#)
[src.software.parse.internal.drive_utility](#)
[src.software.parse.internal.twidlDictGen](#)
[src.software.parse.nlogParser.nlog_triage.hostTimeMarkerTriage](#)
[src.software.parse.nlogParser.nlog_triage.nlogTriageBase](#)
[src.software.parse.nlogParser.nlog_triage.padrTriage](#)
[src.software.parse.nlogParser.nlog_triage.pssDebugTraceTriage](#)
[src.software.parse.nlogParser.telemetry_parsers.Level2Parser](#)
[src.software.parse.nlogParser.telemetry_parsers.nlogEnum](#)
[src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog](#)

[src.software.parse.nlogParser.telemetry_parsers.telemetryHostTimeV1_0](#)
[src.software.parse.nlogParser.test_util.output_log](#)
[src.software.parse.output_log](#)
[src.software.parse.pacmanIC](#)
[src.software.parse.parserDictionaryGen](#)

[src.software.parse.parserSrcUtil](#)
[src.software.parse.parserTwidlGen](#)
[src.software.parse.parserXmlGen](#)
[src.software.parse.structdefParser](#)
[src.software.parse.telemetry_drive](#)
[src.software.parse.telemetryCmd](#)
[src.software.parse.varType](#)
[src.software.threadModuleAPI](#)
[src.software.TSV.configToText](#)
[src.software.TSV.DefragHistoryGrapher](#)
[src.software.TSV.formatTSFiles](#)
[src.software.TSV.generateTSBinaries](#)
[src.software.TSV.genericObjectGUI](#)
[src.software.TSV.loadAndProbeSystem](#)
[src.software.TSV.utilityTSV](#)
[src.software.TSV.visualizeTS](#)
[src.software.userConfigurationProfile](#)
[src.software.utility.templateUtility](#)
[src.software.utilsCommon](#)

Index

[A](#) | [B](#) | [C](#) | [D](#) | [E](#) | [F](#) | [G](#) | [H](#) | [I](#) | [J](#) | [K](#) | [L](#) | [M](#) | [N](#) | [O](#) | [P](#) | [Q](#) | [R](#) | [S](#) | [T](#) | [U](#) | [V](#) | [W](#) | [X](#) | [Z](#)

A

ABRUPT_SHUTDOWN (in module <code>src.software.parse.nlogParser.nlog_triage.pssDebugTraceTriage</code>)	<code>assertD</code> <code>assertE</code> <code>(src.soft</code> <code>method</code>
add_buttons() (<code>src.software.guiDeveloper.GUIDeveloper</code> method)	<code>(sr</code> <code>(sr</code> <code>assertE</code> <code>(src.soft</code> <code>method</code>
add_selectDropDown() (<code>src.software.guiDeveloper.GUIDeveloper</code> method)	<code>(sr</code> <code>(sr</code> <code>assertE</code> <code>(src.soft</code> <code>method</code>
add_selectFolder() (<code>src.software.guiDeveloper.GUIDeveloper</code> method)	<code>(sr</code> <code>(sr</code> <code>assertE</code> <code>(src.soft</code> <code>method</code>
add_typeTextBox() (<code>src.software.guiDeveloper.GUIDeveloper</code> method)	<code>(sr</code> <code>(sr</code> <code>assertE</code> <code>(src.soft</code> <code>method</code>
add_value_labels() (in module <code>src.software.autoAI.mediaPredictionRNN</code>)	<code>(in module src.software.MEP.mediaErrorPredictor)</code> <code>(src.software.autoAI.nlogPrediction.NlogUtils static method)</code>
add_windowLabel() (<code>src.software.guiDeveloper.GUIDeveloper</code> method)	<code>(sr</code> <code>(sr</code> <code>assertF</code> <code>(src.soft</code> <code>method</code>
addApplication() (<code>src.software.userConfigurationProfile.userConfigurationProfile</code> method)	<code>(sr</code> <code>(sr</code> <code>assertF</code> <code>(src.soft</code> <code>method</code>
addClassCleanup() (<code>src.software.axon.axonInterface.Axon_Interface.TestAxonProducer</code> class method)	<code>(src.software.guiTests.AxonRegressionTests class method)</code> <code>(src.software.guiTests.TableRegressionTests class method)</code>
addClassDef() (<code>src.software.parse.parserTwidlGen.twidlParserFileGenerator</code> method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
(src.software.parse.parserXmlGen.xmlFileGenerator method)	
addCleanup() (<code>src.software.axon.axonInterface.Axon_Interface.TestAxonProducer</code> method)	<code>(src.software.guiTests.AxonRegressionTests method)</code> <code>(src.software.guiTests.TableRegressionTests method)</code>
addCustomCommandText() (<code>src.software.dAMP.reportGenerator.ReportGenerator</code> method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addCustomlineText() (<code>src.software.dAMP.reportGenerator.ReportGenerator</code> method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addDefine() (<code>src.software.parse.autoObjects.defineList</code> method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addDoc() (<code>src.software.parse.autoObjects.structDef</code> method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addDocStr() (<code>src.software.parse.autoObjects.dataElement</code> method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addEnumEntry() (<code>src.software.parse.autoObjects.enumValueList</code> method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addEnumerateList() (<code>src.software.dAMP.reportGenerator.ReportGenerator</code> static method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addEnumName() (<code>src.software.parse.autoObjects.enumList</code> method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addImagesRows() (<code>src.software.dAMP.reportGenerator.ReportGenerator</code> static method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addImport() (<code>src.software.parse.parserTwidlGen.parserTwidlGenerator</code> method)	<code>(src.software.parse.parserTwidlGen.twidlParserFileGenerator method)</code>
addIssue() (<code>src.software.JIRA.classJiraIssuesList.jira_issues_lists</code> method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addMathMatrix() (<code>src.software.dAMP.reportGenerator.ReportGenerator</code> static method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addMember() (<code>src.software.parse.autoObjects.structDef</code> method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addMetaTable() (<code>src.software.dAMP.reportGenerator.ReportGenerator</code> method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addNlogEvent()	
(<code>src.software.parse.nlogParser.nlog_triage.hostTimeMarkerTriage.HostTimeMarkerTriage</code> method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
(src.software.parse.nlogParser.nlog_triage.nlogTriageBase.nlogTriageBase method)	
(src.software.parse.nlogParser.nlog_triage.padrTriage.padrTriage method)	
(src.software.parse.nlogParser.nlog_triage.pssDebugTraceTriage.pssDebugTraceTriage method)	
addPDFImages() (<code>src.software.dAMP.reportGenerator.ReportGenerator</code> static method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addPDFImagesRows() (<code>src.software.dAMP.reportGenerator.ReportGenerator</code> static method)	<code>(sr</code> <code>(sr</code> <code>assertG</code> <code>(src.soft</code> <code>method</code>
addr (<code>src.software.JIRA.classGetJiraIssues.get_jira_issues</code> attribute)	
AddSection() (<code>src.software.axon.axonProfile.AxonProfile</code> method)	
addToken() (<code>src.software.parse.parserSrcUtil.fileTokenizer</code> method)	

([src.software.parse.parserSrcUtil.tokenList](#) method),
([src.software.parse.structdefParser.structdefTokenizer](#) method)
[addTypeDef\(\)](#) ([src.software.parse.autoObjects.ctyptedefList](#) method)
[addTypeEqualityFunc\(\)](#) ([src.software.axon.axonInterface.Axon_Interface.TestAxonProducer](#) method)
 ([src.software.guiTests.AxonRegressionTests](#) method),
 ([src.software.guiTests.TableRegressionTests](#) method)
[after\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method),
 ([src.software.MEP.mediaErrorGUI.StartPage](#) method)
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI.StartPage](#) method),
[after_cancel\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.MEP.mediaErrorGUI.StartPage](#) method)
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI.StartPage](#) method),
[after_idle\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.MEP.mediaErrorGUI.StartPage](#) method)
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI.StartPage](#) method),
[aggregateData\(\)](#) ([src.software.MEP.loadAndProbeDrive.loadDrive](#) method)
[aIssuesList](#) ([src.software.JIRA.classGetJiraIssues.get_jira_issues](#) attribute)
[aJiraIssue](#) ([src.software.JIRA.classGetJiraIssues.get_jira_issues](#) attribute),
ALIGNED_4K
([src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.TelemetryV2NlogEventParserL2](#) attribute),
ALIGNED_512
([src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.TelemetryV2NlogEventParserL2](#) attribute),
[allDevCommits\(\)](#) ([src.software.JIRA.classCommonReports.common_reports](#) method),
[allIssues](#) ([src.software.JIRA.classGetJiraIssues.get_jira_issues](#) attribute),
[analysisFrame](#) ([src.software.autoAI.transformCSV.TransformMetaData](#) attribute),
[analysisFrameFormat](#) ([src.software.autoAI.transformCSV.TransformMetaData](#) attribute),
[AnalysisGuide](#) (class in [src.software.JIRA.analysisGuide](#)),
[AnalysisGuide.FirmwareAssert](#) (class in [src.software.JIRA.analysisGuide](#)),
[AnalysisGuide.FirmwareDataForAnalysis](#) (class in [src.software.JIRA.analysisGuide](#)),
[analyzeJira\(\)](#) ([src.software.JIRA.executeJiraMining.executeJiraMining](#) method),
[anchor\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method),
 ([src.software.MEP.mediaErrorGUI.StartPage](#) method)
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI.StartPage](#) method),
[anyObjectToDictionary\(\)](#) (in module [src.software.JIRA.analysisGuide](#))
 ([src.software.DP.preprocessingAPI.preprocessingAPI](#) static method)
 ([src.software.utilsCommon.DictionaryFlatten](#) static method),
[API\(\)](#) (in module [src.software.autoModuleAPI](#)),
 (in module [src.software.dAMP.gatherMeta](#)),
 (in module [src.software.dAMP.reportGenerator](#)),
 (in module [src.software.guiDeveloper](#)),
 (in module [src.software.guiLayouts](#)),
 (in module [src.software.guiTests](#)),
 (in module [src.software.MEP.loadAndProbeDrive](#)),
 (in module [src.software.threadModuleAPI](#)),
 (in module [src.software.TSV.loadAndProbeSystem](#)),
 (in module [src.software.utilsCommon](#)),
[aplReserved](#) ([src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct](#) attribute),
[append\(\)](#) ([src.software.parse.bufdata.BufDataList](#) method),
 ([src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct](#) attribute),
 ([src.software.parse.bufdata.BufDataList](#) method)

[assertT](#)
[\(src.sof](#)
[method](#)
 (sr
 (sr
[assertT](#)
[\(src.sof](#)
[method](#)
 (sr
 (sr
[assertW](#)
[\(src.sof](#)
[method](#)
 (sr
 (sr
[assertW](#)
[\(src.sof](#)
[method](#)
 (sr
 (sr
assisted
aSuppo
attribut
atlassia
method
attribut
(src.sof
method
 (sr
 (sr
auditAv
(src.sof
authorN
attribut
authStr
autoMc
Axon_I
Axon_I
src.soft
AXON
AxonD
AxonD
AxonP
AxonR
AxonU
AxonU

B

[base_str\(\)](#)
[\(src.software.parse.bufdata.BufDataList method\)](#)
[baseFilePath](#)
[\(src.software.guiOneShot.GUIOneShot attribute\)](#)
[bbox\(\)](#)
[\(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method\)](#)
[\(src.software.MEP.mediaErrorGUI.StartPage method\)](#)

[BinDigitList](#)
[\(src.software.par](#)
[\(src.software.parse.parse](#)
[\(src.software.parse.struct](#)
[bindtags\(\)](#)
[\(src.software.MEP](#)
[\(src.software.MEP.media](#)
[\(src.software.TSV.generi](#)

```

(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow
method)
(src.software.TSV.genericObjectGUI.StartPage method)
BeginHtmOutput()(src.software.JIRA.classHtmlGen.htmlGen method)
BeginTable()(src.software.JIRA.classHtmlGen.htmlGen method)
bell()(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow
method)
(src.software.MEP.mediaErrorGUI.StartPage method)
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow
method)
(src.software.TSV.genericObjectGUI.StartPage method)
bertEmbedJira()(src.software.JIRA.jiraEmbedding.jiraEmbedding
method)
BigDataProcessing(class in src.software.autoAI.NNStateBasedProcessing)
binaryPath(src.software.dAMP.gatherMeta.GatherMeta attribute)
bind()(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow
method)
(src.software.MEP.mediaErrorGUI.StartPage method)
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow
method)
(src.software.TSV.genericObjectGUI.StartPage method)
bind_all()
(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow
method)
(src.software.MEP.mediaErrorGUI.StartPage method)
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow
method)
(src.software.TSV.genericObjectGUI.StartPage method)
bind_class()
(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow
method)
(src.software.MEP.mediaErrorGUI.StartPage method)
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow
method)
(src.software.TSV.genericObjectGUI.StartPage method).

```

```

(src.software.TSV.gen
bits_different()(src.software.j
(src.software.parse.bufda
(src.software.parse.bufda
BITSTRUCT_TYPE(src.soft
blRevision(src.software.parse
attribute)
BufData(class in src.software
BufDataList(class in src.soft
buildList()(src.software.parse
(src.software.parse.intel\N
byte_size()(src.software.parse
(src.software.parse.bufda
(src.software.parse.bufda
Bytes(src.software.parse.head
(src.software.parse.intel\N
attribute)
(src.software.parse.intel\N
(src.software.parse.intel\N
(src.software.parse.intel\N
(src.software.parse.intel\N
(attribute)

```

C

```

calculateBandwidth()
(src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher.defragHistoryUtility.static
method)
calculateDrift()(src.software.utilsCommon.FunctionScheduleTimer method)
capFirstLetter()(src.software.parse.autoObjects.outputGenerationHelper method)
(src.software.parse.parserDictionaryGen.parserDictionaryGenerator method)
(src.software.parse.parserTwidlGen.parserTwidlGenerator method)
(src.software.parse.parserXmlGen.parserXmlGenerator method)
CFG(src.software.dAMP.gatherMeta.GatherMeta attribute)
cget()(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)
(src.software.MEP.mediaErrorGUI.StartPage method)
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)
(src.software.TSV.genericObjectGUI.StartPage method)
charHeight(src.software.guiLayouts.dataTablePopulate attribute)
charWidth(src.software.guiLayouts.dataTablePopulate attribute)
check_flag()(src.software.guiDeveloper.GUIDeveloper static method)

```

```

commitColor
commitList(
commitsByD
common_rep
compareDict(
(src.soft
(src.soft
compareObjs(
(src.soft
(src.soft
compileAndF
static method
(src.soft
(src.soft
method)
(src.soft
compressTar(

```

check_stationarity() (src.software.autoAI.mediaPredictionRNN.timeSeriesRNN static method)
 (src.software.MEP.mediaErrorPredictor.MediaErrorPredictor method)
 (src.software.TSV.visualizeTS.visualizeTS method)

checkBinDir() (src.software.TSV.utilityTSV.utilityTSV static method)

checkBooleanOption() (src.software.MEP.loadAndProbeDrive.loadDriveUtility static method)
 (src.software.TSV.utilityTSV.utilityTSV static method)

checkCombine() (src.software.TSV.utilityTSV.utilityTSV static method)

checkDebugOption() (src.software.MEP.loadAndProbeDrive.loadDriveUtility static method)
 (src.software.TSV.utilityTSV.utilityTSV static method)

checkDriveIndex() (src.software.parse.internal.drive_utility.driveList static method)

checkDriveName() (src.software.MEP.loadAndProbeDrive.loadDriveUtility static method)
 (src.software.TSV.utilityTSV.utilityTSV static method)

checkDriveNumber() (src.software.MEP.loadAndProbeDrive.loadDriveUtility static method)
 (src.software.TSV.utilityTSV.utilityTSV static method)

checkEvents()

(src.software.parse.nlogParser.nlog_triage.hostTimeMarkerTriage.HostTimeMarkerTriage method)
 (src.software.parse.nlogParser.nlog_triage.nlogTriageBase.nlogTriageBase method)
 (src.software.parse.nlogParser.nlog_triage.padrTriage.padrTriage method)
 (src.software.parse.nlogParser.nlog_triage.pssDebugTraceTriage.pssDebugTraceTriage method)

checkFWDDir() (src.software.TSV.utilityTSV.utilityTSV static method)

checkIdentifier() (src.software.MEP.loadAndProbeDrive.loadDriveUtility static method)
 (src.software.TSV.utilityTSV.utilityTSV static method)

checkInOutfile() (src.software.TSV.utilityTSV.utilityTSV static method)

checkInputDir() (src.software.TSV.utilityTSV.utilityTSV static method)

checkInputFile() (src.software.TSV.utilityTSV.utilityTSV static method)

checkIterations() (src.software.MEP.loadAndProbeDrive.loadDriveUtility static method)
 (src.software.TSV.utilityTSV.utilityTSV static method)

checkModeSelect() (src.software.TSV.utilityTSV.utilityTSV static method)

checkNumCores() (src.software.TSV.utilityTSV.utilityTSV static method)

CheckObjectHeader() (src.software.parse.intelObjectIdList.ParserObjectMap method)

checkOutFile() (src.software.MEP.loadAndProbeDrive.loadDriveUtility static method)

checkOutfile() (src.software.TSV.utilityTSV.utilityTSV static method)

checkOutpath() (src.software.TSV.utilityTSV.utilityTSV static method)

checkOutputDir() (src.software.MEP.loadAndProbeDrive.loadDriveUtility static method)
 (src.software.TSV.utilityTSV.utilityTSV static method)

checkParseOption() (src.software.TSV.utilityTSV.utilityTSV static method)

checkPythonVersion() (in module src.software.utilsCommon)

checkSubSeqLen() (src.software.TSV.utilityTSV.utilityTSV static method)

checkSum() (in module src.software.utilsCommon)

checkTransformTS() (src.software.TSV.utilityTSV.utilityTSV static method)

Cipher_AES (class in src.software.axon.packageInterface)

Cipher_MODE_CBC() (src.software.axon.packageInterface.Cipher_AES method)

Cipher_MODE_ECB() (src.software.axon.packageInterface.Cipher_AES method)

classToDictionary() (src.software.DP.preprocessingAPI.preprocessingAPI static method)

cleanAndRecreatePath() (in module src.software.utilsCommon)

cleanFileName() (in module src.software.utilsCommon)

cleanList() (src.software.parse.autoObjects.ctypedefList method)
 (src.software.parse.autoObjects.defineList method)
 (src.software.parse.autoObjects.enumList method)
 (src.software.parse.autoObjects.enumValueList method)

compressTarl
config() (src.
 (src.soft
 (src.soft
 (src.soft
configData(s
configFileNa
 (src.soft
configFolder
attribute)
configLocati
configToText
configure() (s
 (src.soft
 (src.soft
 (src.soft
constructSyst
content() (src
contentFile(s
contents() (sr
contentSizeB
(src.software
contentString
continuePars
 (src.soft
controllerInit
attribute)
convertTime(
convertToCa
 (src.soft
 (src.soft
 (src.soft
convertToHC
ConvertToW
(src.software
static method
coreCount
(src.software
attribute)
 (src.soft
attribute
coreSelected
(src.software
attribute)
 (src.soft
attribute
cosineSimila
Cost_Estimat
(src.software
Cost_Estimat
(src.software
countTestCas
method)
 (src.soft
 (src.soft

([src.software.parse.autoObjects.structDefList method](#))
[CleanList\(\)](#) ([src.software.parse.structdefParser.structdefParser method](#))
[cleanSearchString\(\)](#) ([in module src.software.JIRA.executeJiraMining](#))
[clear\(\)](#) ([src.software.parse.parserSrcUtil.fileTokenizer method](#)).
 ([src.software.parse.parserSrcUtil.tokenList method](#)).
 ([src.software.parse.structdefParser.structdefTokenizer method](#))
[client\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
[clipboard_append\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[clipboard_clear\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[clipboard_get\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[close\(\)](#) ([src.software.parse.internal.twidDictGen.OpenFiles method](#))
[cmdnsid](#) ([src.software.parse.telemetry_drive.logDevice attribute](#))
[codeBranchPage](#) ([src.software.dAMP.reportGenerator.ReportGenerator attribute](#))
[collect\(\)](#) ([src.software.guiLayouts.GUILayouts static method](#))
[collectDriveData\(\)](#) ([src.software.MEP.loadAndProbeDrive.loadDrive method](#))
[collectGUI](#) ([class in src.software.guiCommon](#))
[collectionTypes](#) ([src.software.dAMP.reportGenerator.ReportGenerator attribute](#))
[collectTelemetry](#) ([src.software.guiOneShot.GUIOneShot attribute](#))
[colormapwindows\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
[Cols](#) ([src.software.dAMP.iSOM.iSOM attribute](#))
[columnconfigure\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[columnsList](#) ([src.software.autoAI.transformCSV.TransformMetaData attribute](#))
[command\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
[COMMENT_TOKEN](#) ([src.software.parse.parserSrcUtil.parserHelper attribute](#)).
 ([src.software.parse.structdefParser.structdefParser attribute](#))

[cpuId](#)
 ([src.software attribute](#))
[cpuNumber](#)
 ([src.software \(src.soft attribute](#))
 ([src.soft attribute](#))
[create\(\)](#) ([src.create_new_ \(src.soft create_new_ \(src.soft](#))
 ([src.soft create_new_ \(src.soft method](#))
 ([src.soft create_new_ \(src.soft method](#))
 ([src.soft create_new_ static method](#))
 ([src.soft create_new_ \(src.soft method](#))
 ([src.soft create_new_ \(src.soft method](#))
 ([src.soft create_tempo \(src.soft create_tempo method](#))
 ([src.soft create_tempo \(src.soft method](#))
 ([src.soft createContent \(src.soft createCSVFr](#))
 ([src.soft createDataSt \(src.soft createDictFrc](#))
 ([src.software createDir\(\)](#) ([src.soft \(src.soft method](#))
 ([src.soft createDocum \(src.soft createDocum](#))
 ([src.soft createDocum \(src.soft method](#))
 ([src.soft createFilePat \(src.soft CreateFiles\(\)](#)
 ([src.soft \(src.soft createNode\(\)](#)
 ([src.soft createStructC](#))
 ([src.createTar\(\)](#) ([src.createTarball](#))
 ([src.createZIP\(\)](#) ([src.creationTime](#))
 ([src.CSVtoFrame](#))
 ([src.ctrlInitiatedD](#))
 ([src.software attribute](#))
 ([src.soft attribute](#))
 ([src.soft attribute](#))
 ([src.soft attribute](#))

[ctrlInitiatedG](#)
[\(src.software.attribute\)](#)
[\(src.soft.attribute\)](#)
[\(src.soft.attribute\)](#)
[\(src.soft.attribute\)](#)
[ctypedef\(class\)](#)
[ctypedefList](#)
[cUID\(class\)](#)
[customerName](#)
[customerPage](#)

D

[data_x_shadow](#)
[\(src.software.dAMP.iSOM.iSOM attribute\)](#)
[data_y_shadow](#)
[\(src.software.dAMP.iSOM.iSOM attribute\)](#)
[dataArea1EndBlock](#)
[\(src.software.parse.headerTelemetry.NvmeControllerInitiatedLogPageHeader_struct attribute\)](#)
[\(src.software.parse.headerTelemetry.NvmeHostInitiatedLogPageHeader_struct attribute\)](#)
[\(src.software.parse.headerTelemetry.SataControllerInitiatedLogPageHeader_struct attribute\)](#)
[\(src.software.parse.headerTelemetry.SataHostInitiatedLogPageHeader_struct attribute\)](#)
[dataArea2EndBlock](#)
[\(src.software.parse.headerTelemetry.NvmeControllerInitiatedLogPageHeader_struct attribute\)](#)
[\(src.software.parse.headerTelemetry.NvmeHostInitiatedLogPageHeader_struct attribute\)](#)
[\(src.software.parse.headerTelemetry.SataControllerInitiatedLogPageHeader_struct attribute\)](#)
[\(src.software.parse.headerTelemetry.SataHostInitiatedLogPageHeader_struct attribute\)](#)
[dataArea3EndBlock](#)
[\(src.software.parse.headerTelemetry.NvmeControllerInitiatedLogPageHeader_struct attribute\)](#)
[\(src.software.parse.headerTelemetry.NvmeHostInitiatedLogPageHeader_struct attribute\)](#)
[\(src.software.parse.headerTelemetry.SataControllerInitiatedLogPageHeader_struct attribute\)](#)
[\(src.software.parse.headerTelemetry.SataHostInitiatedLogPageHeader_struct attribute\)](#)
[dataCntrlStructPath](#)
[\(src.software.dAMP.gatherMeta.GatherMeta attribute\)](#)
[DataCollect\(\)](#)
[\(src.software.guiOneShot.GUIOneShot method\)](#)
[DataCollectConfig\(\)](#)
[\(src.software.guiOneShot.GUIOneShot method\)](#)
[DATACONTROLCSV](#)
[\(in module src.software.parse.internal.twidlDictGen\)](#)
[DatacontrolGen](#)
[\(class in src.software.datacontrol.dataControlGenMain\)](#)
[DatacontrolGenCSV](#)
[\(class in src.software.datacontrol.dataControlGenMain\)](#)
[DatacontrolGenH](#)
[\(class in src.software.datacontrol.dataControlGenMain\)](#)
[dataControlGenWrapper\(\)](#)
[\(in module src.software.datacontrol.dataControlGenMain\)](#)
[dataDict](#)
[\(src.software.dAMP.gatherMeta.GatherMeta attribute\)](#)

[DefragHistoryGraph](#)
[method\)](#)
[DefragObjectDec](#)
[DefragObjectDec](#)
[deiconify\(\)](#)
[\(src.software\)](#)
[\(src.software.deletecommand\(\).method\)](#)
[\(src.software\)](#)
[\(src.software\)](#)
[\(src.software\)](#)
[DELIMITER_TO](#)
[\(src.software\)](#)
[deviceConfiguration](#)
[deviceSignature](#)
[DFA](#)
[\(src.software\)](#)
[DFG](#)
[\(src.software\)](#)
[dhFilePath](#)
[\(src.software\)](#)
[dict2BulletList\(\)](#)
[dictElement](#)
[\(class\)](#)
[DictionaryFlatten](#)
[dictionaryForward](#)
[DictionaryLower](#)
[DictionaryPrune](#)
[DictionaryReduce](#)
[DictionaryRevers](#)
[dictionaryReverse](#)
[DictionarySetRed](#)
[DictionaryUpper](#)
[digestTextFiles\(\)](#)

dataDictDimension(src.software.dAMP.gatherMeta.GatherMeta attribute)
dataElement(class in src.software.parse.autoObjects)
dataFile(src.software.guiOneShot.GUIOneShot attribute)
dataFileName(src.software.guiOneShot.GUIOneShot attribute)
dataIn(src.software.guiLayouts.dataTablePopulate attribute)
dataIndependantEncoding()
(src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method)
DataLakeMeta(src.software.dAMP.gatherMeta.GatherMeta attribute)
DataParserV2_0(class in src.software.parse.intelTelemetryDataObject)
DataTable()(src.software.guiOneShot.GUIOneShot method)
DataTableConfig()(src.software.guiOneShot.GUIOneShot method)
dataTablePopulate(class in src.software.guiLayouts)
debug(src.software.autoAI.transformCSV.TransformMetaData attribute)
(src.software.dAMP.gatherMeta.GatherMeta attribute)
(src.software.dAMP.iSOM.iSOM attribute)
(src.software.dAMP.reportGenerator.RegressionUnitTestsCases attribute)
(src.software.dAMP.reportGenerator.ReportGenerator attribute)
(src.software.guiLayouts.dataTablePopulate attribute)
(src.software.guiLayouts.GUILayouts attribute)
(src.software.guiOneShot.GUIOneShot attribute)
(src.software.TSV.formatTSFiles.formatTSFiles attribute)
(src.software.TSV.formatTSFiles.formatTSFiles.formatUtility attribute)
(src.software.userConfigurationProfile.userConfigurationProfile attribute)
(src.software.utilsCommon.DictionaryFlatten attribute)
debug()(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method)
(src.software.guiTests.AxonRegressionTests method)
(src.software.guiTests.TableRegressionTests method)
debugComments()(src.software.guiLayouts.GUILayouts static method)
debugOutputLevel(src.software.parse.nlogParser.test_util.output_log.OutputLog attribute)
(src.software.parse.output_log.OutputLog attribute)
debugPrint()(src.software.parse.autoObjects.structDefList method)
DebugPrint()(src.software.parse.nlogParser.test_util.output_log.OutputLog static method)
(src.software.parse.output_log.OutputLog static method)
decodeEventEncoding()(src.software.autoAI.nlogPrediction.NlogTokenizer method)
decomposeDataSets()(src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method)
decrypt()(src.software.axon.packageInterface.Cipher_AES method)
decryptTarball()(src.software.axon.packageInterface.packageInterface method)
DEFAULT_TOKEN(src.software.parse.parserSrcUtil.parserHelper attribute)
(src.software.parse.structdefParser.structdefParser attribute)
defaultHash()(src.software.axon.axonInterface.Axon_Interface method)
defaultLinearModel()(src.software.autoAI.nlogPrediction.NlogWidthPredictor method)
defaultNameCounter(src.software.parse.structdefParser.structdefParser attribute)
defaultTestResult()
(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method)
(src.software.guiTests.AxonRegressionTests method)
(src.software.guiTests.TableRegressionTests method)
defineList(class in src.software.parse.autoObjects)
DefragConfig(class in src.software.guiCommon)
defragHistoryGraph()(src.software.guiLayouts.GUILayouts static method)
DefragHistoryGrapher(class in src.software.TSV.DefragHistoryGrapher)
DefragHistoryGrapher.defragHistoryUtility(class in src.software.TSV.DefragHistoryGrapher)

DigitList(src.soft
(src.software
(src.software
Dim(src.software
DirectAccess(cla
disableQuiet(),sr
(src.software
display_dir(),sr
(src.software
(src.software
display_dir_ARM
display_dir_defra
display_dir_gene
display_dir_RNN
display_windowI
DisplayDebugWi
displayDriveListdisplayReport(sr
distance_euclidea
distance_manhatt
distanceMeasure(
distFuncTest(),(in
doClassCleanups(
method)
(src.software
(src.software
doCleanups()(src
(src.software
(src.software
doExamplePeriod
doExamplePeriod
doExampleWaitL
download()(src.s
downloadAXONI
driveInfoZip()(sr
driveList(class in
dword(src.softwa
attribute)
(src.software
attribute)

E

edit() (src.software.datacontrol.dataControlGenMain DatacontrolGenCSV method)
elasticNetLinearModel() (src.software.autoAI.nlogPrediction.NlogWidthPredictor method)
embeddedPredictor() (src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method)
embeddingNet() (src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method)
embedJira() (src.software.JIRA.executeJiraMining.executeJiraMining method)
EMPTY_TOKEN
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.EventHeader_union
attribute)
enableQuiet() (src.software.parse.nlogParser.test_util.output_log.OutputLog static method)
 (src.software.parse.output_log.OutputLog static method)
enableSilentMode() (src.software.parse.nlogParser.test_util.output_log.OutputLog static
method)
encodeNlogEvents() (src.software.autoAI.nlogPrediction.NlogTokenizer method)
encrypt() (src.software.axon.packageInterface.Cipher_AES method)
encryptTarball() (src.software.axon.packageInterface.packageInterface method)
END_OF_LIST (src.software.parse.parserSrcUtil.parserHelper attribute)
 (src.software.parse.structdefParser.structdefParser attribute)
EndHtmOutput() (src.software.JIRA.classHtmlGen.htmlGen method)
EndTable() (src.software.JIRA.classHtmlGen.htmlGen method)
entryCount (src.software.parse.intelVUTelemetry.intelTelemetryTOC_V2_struct attribute)
enumFormatIdEndMarker
(src.software.parse.nlogParser.telemetry_parsers.nlogEnum.nlogEnumTranslate attribute)
enumFormatIdEndMarkerLen
(src.software.parse.nlogParser.telemetry_parsers.nlogEnum.nlogEnumTranslate attribute)
enumFormatIdMarker
(src.software.parse.nlogParser.telemetry_parsers.nlogEnum.nlogEnumTranslate attribute)
enumFormatIdMarkerLen
(src.software.parse.nlogParser.telemetry_parsers.nlogEnum.nlogEnumTranslate attribute)
enumList (class in src.software.parse.autoObjects)
enumParser()
(src.software.parse.nlogParser.telemetry_parsers.nlogEnum.nlogEnumTranslate method)
enumValueList (class in src.software.parse.autoObjects)
EOL_TOKEN (src.software.parse.parserSrcUtil.fileTokenizer attribute)
 (src.software.parse.parserSrcUtil.tokenList attribute)
 (src.software.parse.structdefParser.structdefTokenizer attribute)
Error() (src.software.parse.nlogParser.test_util.output_log.OutputLog static method)
 (src.software.parse.output_log.OutputLog static method)
errorDetected (src.software.parse.intelVUTelemetryReason.intelTelemetryReasonCode
attribute)
errorDetected()
(src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct method)
errorLevel
(src.software.parse.nlogParser.nlog_triage.hostTimeMarkerTriage.HostTimeMarkerTriage
attribute)
 (src.software.parse.nlogParser.nlog_triage.nlogTriageBase.nlogTriageBase attribute)
 (src.software.parse.nlogParser.nlog_triage.padrTriage.padrTriage attribute)
 (src.software.parse.nlogParser.nlog_triage.pssDebugTrace.pssDebugTraceTriage
 attribute)
event_add() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)
 (src.software.MEP.mediaErrorGUI.StartPage method)
 (src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)
 (src.software.TSV.genericObjectGUI.StartPage method)
event_delete() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)
event_generate
 (src.soft
 (src.soft
 (src.soft
method)
 (src.soft
 (src.soft
 (src.soft
event_info()
method)
 (src.soft
 (src.soft
 (src.soft
EventHeader
src.software.j
EventHeader
src.software.j
eventList
 (src.software
attribute)
EventNum
 (src.software
attribute)
eventPredict
EventTimeSt
 (src.software
EventTimeSt
src.software.j
EventTupleT
 (src.software
example (src
ExampleUsa
excursionDet
 (src.software
excursionDet
 (src.software
method)
execute()
 (src.software
method)
execute_nlog
 (src.soft
 (method)
execute_allJo
executeJiraM
ExecutePhas
ExecuteTask
extractEvent
 (method)
extractFiles
 (extractFilesC
extractFilesF
 (method)
extractInputI
 (method)
extractNlogC

[\(src.software.MEP.mediaErrorGUI.StartPage method\)](#)
[\(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method\)](#)
[\(src.software.TSV.genericObjectGUI.StartPage method\)](#)

[extractNlogT](#)
[extractNlogT](#)
[\(src.soft](#)
[extractTarbal](#)
[extractTelem](#)

F

[fail\(\).\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method\)](#)
[\(src.software.guiTests.AxonRegressionTests method\)](#)
[\(src.software.guiTests.TableRegressionTests method\)](#)
[failIf\(\).\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method\)](#)
[\(src.software.guiTests.AxonRegressionTests method\)](#)
[\(src.software.guiTests.TableRegressionTests method\)](#)
[failIfAlmostEqual\(\)](#)
[\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method\)](#)
[\(src.software.guiTests.AxonRegressionTests method\)](#)
[\(src.software.guiTests.TableRegressionTests method\)](#)
[failIfEqual\(\)](#)
[\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method\)](#)
[\(src.software.guiTests.AxonRegressionTests method\)](#)
[\(src.software.guiTests.TableRegressionTests method\)](#)
[failUnless\(\)](#)
[\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method\)](#)
[\(src.software.guiTests.AxonRegressionTests method\)](#)
[\(src.software.guiTests.TableRegressionTests method\)](#)
[failUnlessAlmostEqual\(\)](#)
[\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method\)](#)
[\(src.software.guiTests.AxonRegressionTests method\)](#)
[\(src.software.guiTests.TableRegressionTests method\)](#)
[failUnlessEqual\(\)](#)
[\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method\)](#)
[\(src.software.guiTests.AxonRegressionTests method\)](#)
[\(src.software.guiTests.TableRegressionTests method\)](#)
[failUnlessRaises\(\)](#)
[\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method\)](#)
[\(src.software.guiTests.AxonRegressionTests method\)](#)
[\(src.software.guiTests.TableRegressionTests method\)](#)
[failure\(\).\(src.software.axon.axonMeta.AXON_Meta method\)](#)
[failureException](#)
[\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer attribute\)](#)
[\(src.software.guiTests.AxonRegressionTests attribute\)](#)
[\(src.software.guiTests.TableRegressionTests attribute\)](#)
[failureModeString](#)
[\(src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct attribute\)](#)
[FaultCostModelForReturnOnInvestment \(class in src.software.estimationROI\)](#)
[fdSize \(src.software.utilsCommon.DictionaryFlatten attribute\)](#)
[feature_intersection\(\)\(src.software.dAMP.iSOM.iSOM static method\)](#)
[Fields \(class in src.software.MEP.mediaErrorGUI\)](#)
[\(class in src.software.TSV.genericObjectGUI\)](#)
[FieldsAttributes \(class in src.software.MEP.mediaErrorGUI\)](#)
[\(class in src.software.TSV.genericObjectGUI\)](#)

[firmwareDFA \(src.softw](#)
[firmwareName \(src.softv](#)
[firstPage \(src.software.d/](#)
[flags \(src.software.parse.](#)
[\(src.software.parse.i](#)
[\(src.software.parse.i](#)
[attribute\)](#)
[flattenJson\(\)\(src.softwar](#)
[\(src.software.JIRA.c](#)
[flattenList\(\)\(in module s](#)
[focus\(\)\(src.software.ME](#)
[\(src.software.MEP.r](#)
[\(src.software.TSV.g](#)
[\(src.software.TSV.g](#)
[focus_displayof\(\)\(src.so](#)
[\(src.software.MEP.r](#)
[\(src.software.TSV.g](#)
[\(src.software.TSV.g](#)
[focus_force\(\)\(src.softwa](#)
[\(src.software.MEP.r](#)
[\(src.software.TSV.g](#)
[\(src.software.TSV.g](#)
[focus_get\(\)\(src.software](#)
[\(src.software.MEP.r](#)
[\(src.software.TSV.g](#)
[\(src.software.TSV.g](#)
[focus_lastfor\(\)\(src.softw](#)
[\(src.software.MEP.r](#)
[\(src.software.TSV.g](#)
[\(src.software.TSV.g](#)
[focus_set\(\)\(src.software](#)
[\(src.software.MEP.r](#)
[\(src.software.TSV.g](#)
[\(src.software.TSV.g](#)
[focusmodel\(\)\(src.softwa](#)
[\(src.software.TSV.g](#)
[forget\(\)\(src.software.MI](#)
[\(src.software.MEP.r](#)
[\(src.software.TSV.g](#)
[\(src.software.TSV.g](#)
[formatDataSets\(\)\(src.sol](#)
[formatNlogTime\(\)\(src.s](#)
[formatTSFiles \(class in s](#)
[formatTSFiles.formatUti](#)
[formatTSFilesAPI\(\)\(src.](#)
[formMetaData\(\)\(src.soft](#)
[\(src.software.guiLay](#)

[FieldsLabels](#) ([src.software.MEP.mediaErrorGUI.FieldsAttributes attribute](#))
 (src.software.TSV.genericObjectGUI.FieldsAttributes attribute)
[FieldsLegendLocation](#) ([src.software.MEP.mediaErrorGUI.FieldsAttributes attribute](#))
 (src.software.TSV.genericObjectGUI.FieldsAttributes attribute)
[FieldsSelectionMode](#) ([src.software.MEP.mediaErrorGUI.FieldsAttributes attribute](#))
 (src.software.TSV.genericObjectGUI.FieldsAttributes attribute)
[file_info\(\)](#) ([src.software.parse.bufdata.BufData method](#))
 (src.software.parse.bufdata.BufDataList method)
 (src.software.parse.bufdata.DirectAccess method)
[fileLocation](#) ([src.software.autoAI.transformCSV.TransformMetaData attribute](#))
[fileName](#) ([src.software.autoAI.transformCSV.TransformMetaData attribute](#))
[filename](#) ([src.software.dAMP.reportGenerator.RegressionUnitTestsCases attribute](#))
[fileName](#) ([src.software.dAMP.reportGenerator.ReportGenerator attribute](#))
[fileSize\(\)](#) ([src.software.autoAI.NNStateBasedProcessing.BigDataProcessing method](#))
[fileTokenizer](#) ([class in src.software.parse.parserSrcUtil](#))
[FillDataStructure\(\)](#)
 (src.software.parse.intelTelemetryDataObject.DataParserV2_0 method)
[finalizeStruct\(\)](#) ([src.software.parse.autoObjects.structDef method](#))
[find_optimal_clusters\(\)](#) ([src.software.JIRA.jiraSimAnalysis.jiraSimAnalysis method](#))
[findAll\(\)](#) ([in module src.software.utilsCommon](#))
[findEUID\(\)](#) ([src.software.parse.intelObjectIdList.ParserObjectMap method](#))
[findEUIDParser\(\)](#) ([src.software.parse.intelObjectIdList.ParserObjectMap method](#))
[findExecutable\(\)](#) ([src.software.axon.axonInterface.Axon_Interface method](#))
 (src.software.MEP.loadAndProbeDrive.loadDriveUtility static method)
 (src.software.TSV.generateTSBinaries.generateTSBinaries.generateUtility static method)
[FindExistingTelemetryINI\(\)](#) ([src.software.guiOneShot.GUIOneShot static method](#))
[findMostRecentUpdate\(\)](#) ([src.software.JIRA.classGetJiraIssues.get_jira_issues method](#))
[findStruct\(\)](#) ([src.software.parse.autoObjects.structDefList method](#))
 (src.software.parse.structdefParser.structdefParser method)

[forwardDirectionDict](#) ([sr frame\(\)](#))
 (src.software.MF
 (src.software.TSV.g
 frameToINI())
 (src.softwa
 freeData())
 (src.software.
 from_buf())
 (src.software.parse.l
 (src.software.parse.l
 from_file())
 (src.software.parse.l
 (src.software.parse.l
[FSM_Action\(\)](#)
 (src.softw
[FSM_Transitions\(\)](#)
 (src.s
 FunctionScheduleTimer
 fwDir ([src.software.dAN
 fwParsersLocation \(\[src.s
 fwRevision \\(\\[src.software\\]\\(#\\)\]\(#\)](#)

G

GatherMeta (class in src.software.dAMP.gatherMeta)	ge
gatherTelemetryData() (src.software.guiDeveloper.GUIDeveloper method)	ge
(src.software.guiOneShot.GUIOneShot method)	(sr
gaussianMixClusters() (src.software.JIRA.jiraSimAnalysis.jiraSimAnalysis method)	ge
generateAllContent() (in module src.software.autoModuleAPI)	ge
generateARMA() (in module src.software.autoModuleAPI)	ge
generateAXONDownload() (in module src.software.autoModuleAPI)	ge
generateAXONUpload() (in module src.software.autoModuleAPI)	ge
generateBinaryDataSet() (in module src.software.autoModuleAPI)	ge
generateDataDictFromConfig() (src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher method)	ge
generateDataSets() (src.software.autoAI.nlogPrediction.NlogDataProcessor method)	ge
generateDataTables() (in module src.software.autoModuleAPI)	ge
generateDecompElemLists() (src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method)	ge

[generateDefragHistory\(\)](#) (in module `src.software.autoModuleAPI`)
[generateDemoData\(\)](#) (`src.software.autoAI.nlogPrediction.NlogDataProcessor` method)
[generateEvent\(\)](#) (`src.software.parse.telemetry_drive.logDevice` method)
[generateFile\(\)](#) (`src.software.parse.parserTwidlGen.twidlParserFileGenerator` method)
 (`src.software.parse.parserXmlGen.xmlFileGenerator` method)
[generateHandbook\(\)](#) (in module `src.software.autoModuleAPI`)
[generateHeader\(\)](#)
 (`src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventTupleTranslate` method)
[generateJIRAClusters\(\)](#) (in module `src.software.autoModuleAPI`)
[generateMP\(\)](#) (`src.software.DP.preprocessingAPI.preprocessingAPI` method)
 (`src.software.TSV.visualizeTS.visualizeTS` method)
[generateNormalizedWindow\(\)](#) (`src.software.autoAI.mediaPredictionRNN.timeSeriesRNN` method)
[generateOptimizer\(\)](#) (`src.software.autoAI.nlogPrediction.NlogPredictor` method)
[generateParseDataSet\(\)](#) (in module `src.software.autoModuleAPI`)
[generatePDFFile\(\)](#) (`src.software.TSV.visualizeTS.visualizeTS` method)
[generatePDFPlots\(\)](#) (`src.software.TSV.visualizeTS.visualizeTS` method)
[generatePlainText\(\)](#) (`src.software.TSV.formatTSFiles.formatTSFiles` method)
[generateRNNLSTM\(\)](#) (in module `src.software.autoModuleAPI`)
[generateString\(\)](#) (in module `src.software.utilsCommon`)
 (`src.software.dAMP.reportGenerator.ReportGenerator` static method)
[generateTelemetryObjects\(\)](#) (`src.software.parse.pacmanIC.pacManIC` method)
[generateTelemetryObjectsAutoParsers\(\)](#) (`src.software.parse.pacmanIC.pacManIC` method)
[generateTelemetryObjectsBuffDict\(\)](#) (`src.software.parse.pacmanIC.pacManIC` method)
[generateTextFromConfigFile\(\)](#) (`src.software.TSV.configToText.configToText` method)
[generateTextFromDict\(\)](#) (`src.software.TSV.configToText.configToText` method)
[generateTimeSeriesGraphs\(\)](#) (in module `src.software.autoModuleAPI`)
[generateTruncatedLists\(\)](#)
 (`src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher.defragHistoryUtility` static method)
 (`src.software.TSV.visualizeTS.visualizeTS.visualizeUtility` static method)
[generateTSBinaries](#) (class in `src.software.TSV.generateTSBinaries`)
[generateTSBinaries.generateUtility](#) (class in `src.software.TSV.generateTSBinaries`)
[generateTSBinariesAPI\(\)](#) (`src.software.TSV.generateTSBinaries.generateTSBinaries` method)
[generateTSPublicationADP\(\)](#) (`src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher` method)
[generateTSPublicationADPinPDF\(\)](#) (`src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher` method)
[generateTSPublicationCDR\(\)](#) (`src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher` method)
[generateTSPublicationCDRinPDF\(\)](#)
 (`src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher` method)
[generateTSPublicationCMD\(\)](#) (`src.software.TSV.visualizeTS.visualizeTS` method)
[generateTSPublicationGUI\(\)](#) (`src.software.TSV.visualizeTS.visualizeTS` method)
[genericLinearPredictor\(\)](#) (`src.software.autoAI.nlogPrediction.NlogWidthPredictor` method)
[GenericObjectAppMainWindow](#) (class in `src.software.MEP.mediaErrorGUI`)
 (class in `src.software.TSV.genericObjectGUI`)
[GenericObjectDecode\(\)](#) (`src.software.guiOneShot.GUIOneShot` method)
[GenericObjectDecodeConfig\(\)](#) (`src.software.guiOneShot.GUIOneShot` method)
[GenericObjectGraph](#) (class in `src.software.guiCommon`)
[genericParseTelemetry\(\)](#) (`src.software.TSV.generateTSBinaries.generateTSBinaries` method)
[genericPredictor\(\)](#) (`src.software.autoAI.mediaPredictionRNN.timeSeriesRNN` method)
 (`src.software.autoAI.nlogPrediction.NlogEventsPredictor` method)
 (`src.software.autoAI.nlogPrediction.NlogParameterPredictor` method)
 (`src.software.autoAI.nlogPrediction.NlogTimePredictor` method)
[genLocalPythonDict\(\)](#) (`src.software.datacontrol.dataControlGenMain.DatacontrolGenCSV` method)
[geometry\(\)](#) (`src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow` method)
 (`src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow` method)

geometry_options (src.software.dAMP.reportGenerator.ReportGenerator .attribute)	Ge
get_all() (src.software.guiDeveloper.GUIDeveloper method)	ge
get_allMeta() (src.software.JIRA.analysisGuide.AnalysisGuide method)	ge
get_direct_access() (src.software.parse.bufdata.DirectAccess method)	ge
get_encryptedText() (src.software.axon.packageInterface.Cipher_AES method)	ge
get_iv() (src.software.axon.packageInterface.Cipher_AES method)	ge
get_jira_issues (class in src.software.JIRA . classGetJiraIssues)	ge
get_key() (src.software.axon.packageInterface.Cipher_AES method)	ge
get_listbox_content() (src.software.MEP.mediaErrorGUI.StartPage method) (src.software.TSV.genericObjectGUI.StartPage method)	ge (sr)
get_tableOptions() (src.software.guiDeveloper.GUIDeveloper static method)	Ge
get_windowInputs() (src.software.guiDeveloper.GUIDeveloper method)	ge
getAlignmentSize() (src.software.parse.autoObjects.dataElement method)	(sr)
getAllLinks() (src.software.JIRA.analysisGuide.AnalysisGuide method)	ge
getAnalysisFrame() (src.software.autoAI.transformCSV.TransformMetaData method)	ge
getAnalysisFrameFormat() (src.software.autoAI.transformCSV.TransformMetaData method)	ge
GetAPIReservedFileName() (src.software.parse.intelObjectList.ParserObjectMap method)	ge
getApplicationDefaults() (src.software.userConfigurationProfile.userConfigurationProfile method)	ge
getApplicationMeta() (src.software.userConfigurationProfile.userConfigurationProfile method)	ge
getApplications() (src.software.dAMP.gatherMeta.GatherMeta method) (src.software.userConfigurationProfile.userConfigurationProfile static method)	ge (sr)
getARMAFigures() (src.software.dAMP.gatherMeta.GatherMeta method)	ge
getAssistedFigures() (src.software.dAMP.gatherMeta.GatherMeta method)	ge
getAsyncManager() (src.software.parse.telemetry_drive.logDevice method)	ge
getAvailableCPUCount() (in module src.software.threadModuleAPI)	
getAvailableTemp() (src.software.datacontrol.dataControlGenMain.DatacontrolGenCSV method)	
getAXONIDs() (src.software.guiDeveloper.GUIDeveloper static method)	ge
getBaseSize() (src.software.parse.varType.ghsCvarType method) (src.software.parse.varType.varType method)	ge ge
getBitCount() (src.software.parse.autoObjects.dataElement method)	ge
getBlockSize() (src.software.parse.intelVUTelemetry.tobjectCatalogEntry_struct method) (src.software.parse.intelVUTelemetry.tocObjectEntryV2_struct method)	ge
getboolean() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method) (src.software.MEP.mediaErrorGUI.StartPage method)	ge
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	
(src.software.TSV.genericObjectGUI.StartPage method)	
getbusinessUnit() (src.software.container.basicTypes.UID_Org method)	Ge
getByteSize() (src.software.container.indirection.OneAPIIndirection method) (src.software.parse.intelVUTelemetry.tobjectCatalogEntry_struct method)	me
 (src.software.parse.intelVUTelemetry.tocObjectEntryV2_struct method)	ge
getBytesSize() (in module src.software.utilsCommon)	(sr)
getBytesStr() (src.software.parse.varType.ghsCvarType method) (src.software.parse.varType.varType method)	ge me
getCapName() (src.software.parse.autoObjects.objectDefine method)	ge
getClidata() (src.software.parse.headerTelemetry.NvmeControllerInitiatedLogPageHeader_struct method) (src.software.parse.headerTelemetry.NvmeHostInitiatedLogPageHeader_struct method)	me
 (src.software.parse.headerTelemetry.SataControllerInitiatedLogPageHeader_struct method)	ge
 (src.software.parse.headerTelemetry.SataHostInitiatedLogPageHeader_struct method)	me
getClasstoDictionary() (src.software.dAMP.gatherMeta.GatherMeta method)	
getColumnList() (src.software.autoAI.transformCSV.TransformMetaData method)	
getCommitFile() (src.software.JIRA.classSupportFiles.support_file method)	
GetConfig() (src.software.guiOneShot.GUIOneShot method)	
GetContentReport() (src.software.guiOneShot.GUIOneShot method)	
getCoreSelected() (src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelect_struct method)	

(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV2_struct method)	ge
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV4_struct method)	(sr)
getCpuId()	me
(src.software.parse.intelTelemetryDataObject.intelTelemetryDataObjectHeaderV2_0_struct method)	
getCPUNumber()	
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_4_struct method)	
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_struct method)	
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V2_struct method)	
getCTypeType()	
(src.software.parse.autoObjects.dataElement method)	
getData()	
(src.software.container.basicTypes.UID_Application method)	
(src.software.container.basicTypes.UID_Data method)	
getDataAreaLastBlock()	
(src.software.parse.headerTelemetry.NvmeControllerInitiatedLogPageHeader_struct method)	Ge
(src.software.parse.headerTelemetry.NvmeHostInitiatedLogPageHeader_struct method)	ge
(src.software.parse.headerTelemetry.SataControllerInitiatedLogPageHeader_struct method)	ge
(src.software.parse.headerTelemetry.SataHostInitiatedLogPageHeader_struct method)	ge
getDataAsArray()	
(src.software.guiDeveloper.GUIDeveloper static method)	
(src.software.guiLayouts.dataTablePopulate static method)	ge
getDataDic()	
(src.software.guiDeveloper.GUIDeveloper static method)	ge
(src.software.guiLayouts.dataTablePopulate static method)	ge
GetDataFile()	
(src.software.guiOneShot.GUIOneShot method)	ge
getDataLakeMeta()	
(src.software.dAMP.gatherMeta.GatherMeta method)	ge
getDataOffsetAndSize()	ge
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_4_struct method)	ge
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_struct method)	ge
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V2_struct method)	ge
getDataSize()	
(src.software.parse.intelVUTelemetry.intelTelemetryTOC_V1_struct method)	ge
(src.software.parse.intelVUTelemetry.intelTelemetryTOC_V2_struct method)	ge
getDateFromFileName()	
(src.software.autoAI.nlogPrediction.NlogUtils static method)	ge
getDateFromName()	
(src.software.DP.preprocessingAPI.preprocessingAPI static method)	ge
(src.software.TSV.utilityTSV.utilityTSV static method)	ge
getDateParser()	
(src.software.autoAI.transformCSV.TransformMetaData static method)	(sr)
getTimeFormatString()	
(in module src.software.utilsCommon)	
getDebug()	
(src.software.TSV.configToText.configToText method)	
(src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher method)	
(src.software.TSV.formatTSFiles.formatTSFiles method)	
(src.software.TSV.generateTSBinaries.generateTSBinaries method)	
(src.software.TSV.visualizeTS.visualizeTS method)	
getDebugInfo()	
(src.software.JIRA.analysisGuide.AnalysisGuide.FirmwareAssert method)	(sr)
getDecoderCol()	me
(src.software.estimationROI.NASA_Data method)	ge
getDecoderRow()	
(src.software.estimationROI.NASA_Data method)	ge
getDefault()	
(src.software.guiCommon.dictElement method)	ge
getDefaultAccessCLI()	
(src.software.userConfigurationProfile.userConfigurationProfile method)	
getDefaultHomeSave()	
(src.software.userConfigurationProfile.userConfigurationProfile method)	ge
getDefaultMemberName()	
(src.software.parse.autoObjects.structDef method)	ge
getDefaultSubstructName()	
(src.software.parse.autoObjects.objectDefine method)	ge
(src.software.parse.autoObjects.structDef method)	Ge
getDescData()	
(src.software.parse.autoObjects.dataElement method)	(sr)
getDeviceConfiguration()	
(src.software.dAMP.gatherMeta.GatherMeta method)	ge
getDeviceSignature()	
(src.software.dAMP.gatherMeta.GatherMeta method)	ge
getDict()	
(src.software.datacontrol.dataControlGenMain.DatacontrolGenCSV method)	
getDictionary()	
(in module src.software.parse.internal.twidlDictGen)	
(src.software.parse.internal.twidlDictGen.twidlDictGenerator method)	ge
(src.software.parse.pacmanIC.pacManIC method)	ge
getDictionaryEntry()	
(src.software.parse.intelVUTelemetry.telemetryToObjectCatalogEntry_struct method)	

(src.software.parse.intelVUTelemetry.telemetryTOCObjectEntryV2_struct method)	
getDicts()(src.software.datacontrol.dataControlGenMain.DatacontrolGenCSV method)	ge
getDictSet()(src.software.guiCommon.collectGUI method)	
getDisplaySize()(src.software.parse.autoObjects.structDef method)	
getDoc()(src.software.parse.autoObjects.dataElement method)	ge
getdouble()(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	ge
(src.software.MEP.mediaErrorGUI.StartPage method)	ge
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	ge
(src.software.TSV.genericObjectGUI.StartPage method)	ge
getDrive()(src.software.parse.telemetry_drive.logDevice method)	Ge
getDriveFromList()(src.software.parse.internal.drive_utility.driveList static method)	ge
getDwordCount()	ge
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventHeader_union method)	ge
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventTimeStamp_union static method)	(sr ge
getEntry()(src.software.parse.intelVUTelemetry.intelTelemetryTOC_V1_struct method)	ge
(src.software.parse.intelVUTelemetry.intelTelemetryTOC_V2_struct method)	ge
(src.software.parse.pacmanIC.pacManIC method)	
getEntryCount()(src.software.parse.intelVUTelemetry.intelTelemetryTOC_V1_struct method)	ge
(src.software.parse.intelVUTelemetry.intelTelemetryTOC_V2_struct method)	ge
getEnumEntry()(src.software.parse.autoObjects.enumValueList method)	
getEnumSize()(src.software.parse.varType.ghsCvarType method)	
(src.software.parse.varType.varType method)	ge
GetEUIDFileName()(src.software.parse.intelObjectIdList.ParserObjectMap method)	ge
GetEUIDHumanName()(src.software.parse.intelObjectIdList.ParserObjectMap method)	ge
getEventNum()(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventHeader_union method)	Ge
GetEventNumber()(src.software.parse.intelObjectIdList.ParserObjectMap method)	
getEventTupleList()	ge
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_NlogEventPoolParser method)	
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_TelemetryV2NlogEventParserL2 method)	
getExceptionInfo()	
(src.software.threadModuleAPI.MassiveParallelismSingleFunctionManyParameters method)	
getExecutionTime()	
(src.software.threadModuleAPI.MassiveParallelismSingleFunctionManyParameters method)	
getFamily()(src.software.parse.internal.drive_utility.testDrive method)	
getFamilyID()(src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct method)	ge me
getFieldDocString()(src.software.JIRA.jiraEmbedding.jiraEmbedding method)	ge
getFieldDocument()(src.software.JIRA.jiraEmbedding.jiraEmbedding method)	ge
getFieldMessage()(src.software.JIRA.jiraEmbedding.jiraEmbedding method)	
getFieldSet()(src.software.JIRA.jiraEmbedding.jiraEmbedding method)	ge
getFile()(src.software.parse.intelTelemetryDataObject.DataParserV2_0 method)	ge
getFileFormats()(in module src.software.utilsCommon)	ge
getFileNameUTCTime()(in module src.software.utilsCommon)	
getFirmwareDFA()(src.software.dAMP.gatherMeta.GatherMeta method)	
getFlatDictionary()(src.software.utilsCommon.DictionaryFlatten method)	
getFlattenDictionary()(in module src.software.JIRA.analysisGuide)	ge
(src.software.dAMP.gatherMeta.GatherMeta method)	ge
(src.software.utilsCommon.DictionaryFlatten method)	ge
getFormatDictionaryKey()	gh
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventHeader_union method)	gl
getGroup()(src.software.container.basicTypes.UID_Org method)	gn
getHandbookInfo()(src.software.JIRA.analysisGuide.AnalysisGuide method)	go
getHashFromContentMetaData()(src.software.axon.axonInterface.Axon_Interface method)	go

getHeaderFromFile()	(src.software.autoAI.transformCSV.TransformMetaData method)	gr
getHumanName()	(src.software.parse.intelVUTelemetry.intelLogObjectInformation_struct method)	gr
	(src.software.parse.intelVUTelemetry.intelLogObjectInformation_V_1_4_struct method)	
	(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_4_struct method)	
	(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_struct method)	
	(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V2_struct method)	gr
getIdentity()	(src.software.axon.axonInterface.Axon_Interface method)	gr
getIdNumber()	(src.software.parse.intelTelemetryDataObject.intelTelemetryDataObjectHeaderV2_0_struct method)	gr
getIgnoreToken()	(src.software.parse.parserSrcUtil.parserHelper method)	gr
	(src.software.parse.structdefParser.structdefParser method)	
getInfo()	(src.software.dAMP.gatherMeta.GatherMeta method)	
getINIFileClass()	(src.software.JIRA.analysisGuide.AnalysisGuide method)	gr
getINIFileDictionary()	(src.software.JIRA.analysisGuide.AnalysisGuide method)	gr
getInputCommandLine()	(src.software.userConfigurationProfile.userConfigurationProfile static method)	gr
getInputs()	(src.software.guiCommon.dictElement method)	
getInt()	(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	gr
	(src.software.MEP.mediaErrorGUI.StartPage method)	
	(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	
	(src.software.TSV.genericObjectGUI.StartPage method)	
getInterfaceHeader()	(src.software.parse.telemetry_drive.logDevice method)	gr
getInterfaceLogPageId()	(src.software.parse.headerTelemetry.TelemetryLogPageHeader_union method)	me
getInterfaceTelemetryHeaderStruct()		gr
	(src.software.parse.headerTelemetry.TelemetryLogPageHeader_union method)	(sr)
getIterations()	(src.software.TSV.generateTSBinaries.generateTSBinaries method)	gr
getJIRA()	(src.software.dAMP.gatherMeta.GatherMeta method)	me
getJiraLabels()	(src.software.JIRA.jiraEmbedding.jiraEmbedding static method)	gr
getKey()	(src.software.guiCommon.dictElement method)	(sr)
getLastBlock()	(src.software.parse.headerTelemetry.NvmeControllerInitiatedLogPageHeader_struct method)	gr
	(src.software.parse.headerTelemetry.NvmeHostInitiatedLogPageHeader_struct method)	
	(src.software.parse.headerTelemetry.SataControllerInitiatedLogPageHeader_struct method)	
	(src.software.parse.headerTelemetry.SataHostInitiatedLogPageHeader_struct method)	gr
getLastToDate()	(src.software.JIRA.classJiraIssuesList.issue_utility method)	me
getLCCPS()	(src.software.estimationROI.NASA_Data method)	gr
getLCFPS()	(src.software.estimationROI.NASA_Data method)	gr
getListFailedObjects()	(src.software.parse.pacmanIC.pacManIC method)	
getListPassedObjects()	(src.software.parse.pacmanIC.pacManIC method)	
getLiteral()	(src.software.dAMP.reportGenerator.ReportGenerator static method)	gr
getLogByteSize()	(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelect_struct method)	gr
	(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV2_struct method)	me
	(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV4_struct method)	
getLogId()	(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelect_struct method)	gr
	(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV2_struct method)	
	(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV4_struct method)	gr
getLogName()	(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelect_struct method)	gr
	(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV2_struct method)	
	(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV4_struct method)	gr
getLogNameBase()	(src.software.parse.telemetry_drive.logDevice method)	gr
getLogPage()	(src.software.parse.telemetry_drive.logDevice method)	
getLogPage0()	(src.software.parse.telemetry_drive.logDevice method)	

getMajor() (<code>src.software.container.basicTypes.UID_Application</code> method)	gr
(<code>src.software.container.basicTypes.UID_Data</code> method)	
(<code>src.software.container.basicTypes.VersionTracking</code> method)	
getMajorVersion() (<code>src.software.parse.autoObjects.objectDefine</code> method)	gr
(<code>src.software.parse.autoObjects.structDef</code> method)	
(<code>src.software.parse.intelTelemetryDataObject.intelTelemetryDataObjectHeaderV2_0_struct</code> method)	gr
(<code>src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_4_struct</code> method)	gr
(<code>src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_struct</code> method)	me
(<code>src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V2_struct</code> method)	me
(<code>src.software.parse.intelVUTelemetry.intelTelemetryTOC_V1_struct</code> method)	gr
(<code>src.software.parse.intelVUTelemetry.intelTelemetryTOC_V2_struct</code> method)	gr
(<code>src.software.parse.intelVUTelemetry.intelTelemetryTOC_VersionStruct</code> method)	gr
(<code>src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct</code> method)	gr
getMaxAIU() (<code>src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher</code> . <code>defragHistoryUtility</code> static method)	gr
getMemberList() (<code>src.software.parse.autoObjects.structDef</code> method)	gr
getMenu() (<code>src.software.guiDeveloper.GUIDeveloper</code> method)	gr
	gu
	GI

H

handbookInfo (<code>src.software.JIRA.analysisGuide.AnalysisGuide</code> attribute)	hostMetaData (<code>src.soft</code>
HandbookMeta (class in <code>src.software.JIRA.analysisGuide</code>)	hostTimeFlag <code>src.softw</code>
header	HostTimeMa <code>src.softw</code>
(<code>src.software.parse.intelTelemetryDataObject.intelTelemetryDataObjectHeaderV2_0_union</code> attribute)	src.softw <code>htmlGen (cl</code>
(<code>src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventHeader_union</code> attribute)	htmlOutput (s <code>attribute)</code>
HexDigitList (<code>src.software.parse.parserSrcUtil.numericParser</code> attribute)	(src.soft
(<code>src.software.parse.parserSrcUtil.parserHelper</code> attribute)	htmlOutputFi <code>(src.softw</code>
(<code>src.software.parse.structdefParser.structdefParser</code> attribute)	(src.soft
hollowShell() (in module <code>src.software.axon.axonInterface</code>)	hUID (class i <code>(src.softw</code>
(in module <code>src.software.axon.axonMeta</code>)	hybridGetBin <code>(src.softw</code>
(in module <code>src.software.axon.packageInterface</code>)	hybridGetTel <code>(src.softw</code>
homeDir (<code>src.software.dAMP.reportGenerator.RegressionUnitTestsCases</code> attribute)	hyperlink() (s <code>method)</code>
host() (<code>src.software.axon.axonMeta.AXON_Meta</code> static method)	
HOST_TIME_SET_TOKEN	
(<code>src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventHeader_union</code> attribute)	

I

iconbitmap() ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow](#) method)
iconify() ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow](#) method)
iconmask() ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow](#) method)
iconname() ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow](#) method)
iconphoto() ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow](#) method)
iconposition() ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow](#) method)
iconwindow() ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow](#) method)
id() ([src.software.axon.axonInterface.Axon_Interface](#).[TestAxonProducer](#) method).
 ([src.software.guiTests.AxonRegressionTests](#) method)
 ([src.software.guiTests.TableRegressionTests](#) method)
IDENTIFIER_TOKEN ([src.software.parse.parserSrcUtil](#).[fileTokenizer](#) attribute).
 ([src.software.parse.parserSrcUtil](#).[parserHelper](#) attribute).
 ([src.software.parse.parserSrcUtil](#).[tokenList](#) attribute)
 ([src.software.parse.structdefParser](#).[structdefParser](#) attribute)
 ([src.software.parse.structdefParser](#).[structdefTokenizer](#) attribute).
identifyLatestRunDestDirectory().
([src.software.parse.internal.twidlDictGen](#).[twidlDictGenerator](#) method)
idNumber
([src.software.parse.intelTelemetryDataObject](#).[intelTelemetryDataObjectHeaderV2_0_struct](#) attribute)
IEEE_OUI_id
([src.software.parse.headerTelemetry](#).[NvmeControllerInitiatedLogPageHeader_struct](#) attribute)
 ([src.software.parse.headerTelemetry](#).[NvmeHostInitiatedLogPageHeader_struct](#) attribute)
 ([src.software.parse.headerTelemetry](#).[SataControllerInitiatedLogPageHeader_struct](#) attribute)
 ([src.software.parse.headerTelemetry](#).[SataHostInitiatedLogPageHeader_struct](#) attribute)
image_names() ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method)
 ([src.software.MEP.mediaErrorGUI](#).[StartPage](#) method)
 ([src.software.TSV.genericObjectGUI](#).[GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI](#).[StartPage](#) method).
image_types() ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow](#) method).
 ([src.software.MEP.mediaErrorGUI](#).[StartPage](#) method)
 ([src.software.TSV.genericObjectGUI](#).[GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI](#).[StartPage](#) method).
implement() ([src.software.datacontrol](#).[dataControlGenMain](#).[DatacontrolGenCSV](#) method).
inBCH ([src.software.JIRA.classJiraIssuesList](#).[jira_issues_lists](#) attribute)
inClosed ([src.software.JIRA.classJiraIssuesList](#).[jira_issues_lists](#) attribute)
inDevVal ([src.software.JIRA.classJiraIssuesList](#).[jira_issues_lists](#) attribute)
inFixed ([src.software.JIRA.classJiraIssuesList](#).[jira_issues_lists](#) attribute)
info() ([src.software.MEP.mediaErrorGUI](#).[StartPage](#) method)
 ([src.software.TSV.genericObjectGUI](#).[StartPage](#) method).
information
([src.software.parse.nlogParser](#).[nlog_triage.hostTimeMarkerTriage](#).[HostTimeMarkerTriage](#) attribute)
 ([src.software.parse.nlogParser](#).[nlog_triage.nlogTriageBase](#).[nlogTriageBase](#) attribute)
 ([src.software.parse.nlogParser](#).[nlog_triage.padrTriage](#).[padrTriage](#) attribute).
isDefine() ([src](#))
isDouble() ([src](#))
 ([src.soft](#))
isDoubleDou
 ([src.soft](#))
isDriveAsser
isEmpty()
([src.software](#) method)
isEndofList()
 ([src.soft](#))
 ([src.soft](#))
isEnum() ([src](#))
 ([src.soft](#))
 ([src.soft](#))
 ([src.soft](#))
isFloat() ([src](#))
 ([src.soft](#))
isFloatType()
 ([src.soft](#))
isHiLogTest(
IsHostInitiate
([src.software](#) method)
 ([src.soft](#))
 ([src.soft](#))
 ([src.soft](#))
 ([src.soft](#))
 ([src.soft](#))
 ([src.soft](#))
isHostTimeS
 ([src.software](#) method)
isHostTimeS
 ([src.software](#) method)
isInline() ([src](#))
isInteger() ([src](#))
 ([src.soft](#))
isIntegerKey
 ([src.soft](#))
isIntegerOrM
 ([src.soft](#))
isIntegerOrM
 ([src.soft](#))
isIntegerType
 ([src.soft](#))
 ([src.soft](#))
isLong() ([src](#))
 ([src.soft](#))
isLongChar()
 ([src.soft](#))
isLongLong()
 ([src.soft](#))
isNlogObject

([src.software.parse.nlogParser.nlog_triage.pssDebugTraceTriage.pssDebugTraceTriage_attribute](#))
[Information\(\)](#) ([src.software.parse.nlogParser.test_util.output_log.OutputLog static method](#))
 ([src.software.parse.output_log.OutputLog static method](#))
[inFPV](#) ([src.software.JIRA.class.JiraIssuesList.jira_issues_lists attribute](#))
[inHold](#) ([src.software.JIRA.class.JiraIssuesList.jira_issues_lists attribute](#))
[init_frame\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
[initializeDataDicts\(\)](#)
([src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher.defragHistoryUtility static method](#))
[inputLocation](#) ([src.software.guiOneShot.GUIOneShot attribute](#))
[inputsAPI\(\)](#) ([in module src.software.MEP.loadAndProbeDrive](#))
 ([in module src.software.TSV.loadAndProbeSystem](#))
[IntelIMASCommand\(\)](#) ([src.software.MEP.loadAndProbeDrive.loadDriveUtility method](#))
 ([src.software.TSV.generateTSBinaries.generateTSBinaries.generateUtility method](#))
[intelLogObjectInformation_struct](#) ([class in src.software.parse.intelVUTelemetry](#))
[intelLogObjectInformationV_1_4_struct](#) ([class in src.software.parse.intelVUTelemetry](#))
[intelmasGetBinary\(\)](#) ([src.software.TSV.generateTSBinaries.generateTSBinaries method](#))
[intelmasGetTelemetry\(\)](#) ([src.software.TSV.generateTSBinaries.generateTSBinaries method](#))
[intelReserved](#)
([src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct attribute](#))
[intelTelemetryDataAreaValidation_union](#) ([class in src.software.parse.intelVUTelemetry](#))
[intelTelemetryDataObjectHeaderV2_0_struct](#) ([class in src.software.parse.intelTelemetryDataObject](#))
[intelTelemetryDataObjectHeaderV2_0_union](#) ([class in src.software.parse.intelTelemetryDataObject](#))
[intelTelemetryObjectHeader_union](#) ([class in src.software.parse.intelVUTelemetry](#))
[intelTelemetryObjectHeader_V1_4_struct](#) ([class in src.software.parse.intelVUTelemetry](#))
[intelTelemetryObjectHeader_V1_struct](#) ([class in src.software.parse.intelVUTelemetry](#))
[intelTelemetryObjectHeader_V2_struct](#) ([class in src.software.parse.intelVUTelemetry](#))
[intelTelemetryReasonCode](#) ([class in src.software.parse.intelVUTelemetryReason](#))
[intelTelemetryReasonV1_0_Struct](#) ([class in src.software.parse.intelVUTelemetryReason](#))
[intelTelemetryTOC_union](#) ([class in src.software.parse.intelVUTelemetry](#))
[intelTelemetryTOC_V1_struct](#) ([class in src.software.parse.intelVUTelemetry](#))
[intelTelemetryTOC_V2_struct](#) ([class in src.software.parse.intelVUTelemetry](#))
[intelTelemetryTOC_VersionStruct](#) ([class in src.software.parse.intelVUTelemetry](#))
[internalGetBinary\(\)](#) ([src.software.TSV.generateTSBinaries.generateTSBinaries method](#))
[internalGetTelemetry\(\)](#) ([src.software.TSV.generateTSBinaries.generateTSBinaries method](#))
[is_data_selected_from_fields\(\)](#) ([src.software.guiCommon.GenericObjectGraph static method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[isAllFFs\(\)](#) ([src.software.parse.bufdata.BufData method](#))
 ([src.software.parse.bufdata.BufDataList method](#))
 ([src.software.parse.bufdata.DirectAccess method](#))
[isArray\(\)](#) ([src.software.parse.autoObjects.dataElement method](#))
[isBitField\(\)](#) ([src.software.parse.autoObjects.dataElement method](#))
[isBitFieldType\(\)](#) ([src.software.parse.autoObjects.structDef method](#))
[isBool\(\)](#) ([src.software.parse.autoObjects.dataElement method](#))
 ([src.software.parse.varType.ghsCvarType method](#))
 ([src.software.parse.varType.varType method](#))
[isChar\(\)](#) ([src.software.parse.autoObjects.dataElement method](#))
 ([src.software.parse.varType.ghsCvarType method](#))
 ([src.software.parse.varType.varType method](#))
[isNumber\(\)](#) ()
 ([src.soft](#)
 ([src.soft](#)
[iSOM](#) ([class](#))
[isPeriodRead](#)
[isPointer\(\)](#) ([src](#)
 ([src.soft](#)
 ([src.soft](#)
[isPossible\(\)](#) ()
[isPossibleBit](#)
[isQualifier\(\)](#) ()
[isShort\(\)](#) ([src](#)
 ([src.soft](#)
[isShortChar\(\)](#) ()
 ([src.soft](#)
[isShortShort\(\)](#) ()
 ([src.soft](#)
[isSigned\(\)](#) ([src](#)
 ([src.soft](#)
 ([src.soft](#)
[isSignedChar\(\)](#) ()
 ([src.soft](#)
[isSpecialId\(\)](#) ()
[isSpecialNaN](#) ()
 ([src.soft](#)
 ([src.soft](#)
 ([src.soft](#)
[isStruct\(\)](#) ([src](#)
 ([src.soft](#)
[isStructOrUn](#)
 ([issue.utility](#))
[issueDictionary](#)
[issueList](#) ([src](#))
[issuesByDev](#)
[isTelemetryV](#)
[isTimeAdjust](#) ()
 ([src.software](#)
 method)
[isTypedef\(\)](#) ()
[isUnion\(\)](#) ([src](#)
 ([src.soft](#)
[isValid\(\)](#) ()
 ([src.software](#)
 method)
[isValidationC](#)
[isValidIdentit](#) ()
 ([src.soft](#)
[isVarTypeTol](#) ()
 ([src.soft](#)
[IsVersion1\(\)](#) ()
 ([src.software](#)
 method)
 ([src.soft](#)
 method)
 ([src.soft](#)
 method)

isControllerInitiated() (src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct method)	(src.soft method)
	isWallClockF (src.software method)

J

jira_issue (class in src.software.JIRA.classJiraIssues) jira_issues_lists (class in src.software.JIRA.classJiraIssuesList) jiraCollect() (src.software.JIRA.classGetJiraIssues.get_jira_issues method)	jiraEmbedding (class in src.software.JIRA.jiraEmbedding) jiraLogin() (src.software.JIRA.classGetJiraIssues.get_jira_issue method) jiraSimAnalysis (class in src.software.JIRA.jiraSimAnalysis) jqStr (src.software.JIRA.classGetJiraIssues.get_jira_issue attribute)
---	---

K

keyAndUpdated (src.software.JIRA.classGetJiraIssues.get_jira_issues attribute) keyExists() (src.software.guiCommon.collectGUI method) keys() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method) (src.software.MEP.mediaErrorGUI.StartPage method) (src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method) (src.software.TSV.genericObjectGUI.StartPage method) .	KEYWORD_TOKEN (src.software.parse.parserSrcI attribute) (src.software.parse.struct attribute) KeyWordList (src.software.parse.structdefP attribute) kMeansClustering() (src.software.JIRA.jiraSimAn method)
--	---

L

lassoLinearModel() (src.software.autoAI.nlogPrediction.NlogWidthPredictor method)	1
last_file (src.software.parse.bufdata.BufData attribute)	1
(src.software.parse.bufdata.BufDataList attribute)	1
(src.software.parse.bufdata.DirectAccess attribute)	1
last_file_stat (src.software.parse.bufdata.BufData attribute)	1
(src.software.parse.bufdata.BufDataList attribute)	1
(src.software.parse.bufdata.DirectAccess attribute)	1
LCCPS_CostMean() (src.software.estimationROI.FaultCostModelForReturnOnInvestment method)	1
LCCPS_CostTotal() (src.software.estimationROI.FaultCostModelForReturnOnInvestment method)	1
LCFPS_Percentage() (src.software.estimationROI.FaultCostModelForReturnOnInvestment method)	1
LearnMax (src.software.dAMP.iSOM.iSOM attribute)	
Level2Parser (class in src.software.parse.nlogParser.telemetry_parsers.Level2Parser)	1
lift() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	(
(src.software.MEP.mediaErrorGUI.StartPage method)	2
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	1
(src.software.TSV.genericObjectGUI.StartPage method)	
linearOrder (class in src.software.mp.order)	1
listbox (src.software.MEP.mediaErrorGUI.StartPage attribute)	1
(src.software.TSV.genericObjectGUI.StartPage attribute)	

```
listBox2.(src.software.MEP.mediaErrorGUI.StartPage attribute)
         (src.software.TSV.genericObjectGUI_StartPage attribute)
loadAll().(src.software.JIRA.analysisGuide.AnalysisGuide method)
loadAndProbeDrive().(src.software.guiDeveloper.GUIDeveloper method)
loadCache().(src.software.JIRA.analysisGuide.AnalysisGuide method)
loadCacheAndSearch().(src.software.JIRA.analysisGuide.AnalysisGuide method)
loadConfigIntoDict().(src.software.DP.preprocessingAPI.preprocessingAPI static method)
loadCSVIntoDict().(src.software.DP.preprocessingAPI.preprocessingAPI static method)
loadData().(src.software.parse.nlogParser.telemetry_parsers.telemetryHostTimeV1_0.hostTimeFlags
method)
         (src.software.parse.nlogParser.telemetry_parsers.telemetryHostTimeV1_0.telemetryHostTimeV1_0
method)
loadDataDict().(src.software.DP.preprocessingAPI.preprocessingAPI method)
loadDictIntoConfig().(src.software.DP.preprocessingAPI.preprocessingAPI static method)
loadDictIntoDataFrames().(src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method)
loadDrive(class in src.software.MEP.loadAndProbeDrive)
```

M

[main\(\)](#) (in module `src.software.autoAI.mediaPredictionRNN`)
[\(in module src.software.autoAI.nlogPrediction\)](#)
[\(in module src.software.autoAI.NNStateBasedProcessing\)](#)
[\(in module src.software.autoAI.transformCSV\)](#)
[\(in module src.software.autoModuleAPI\)](#)
[\(in module src.software.axon.axonInterface\)](#)
[\(in module src.software.axon.axonMeta\)](#)
[\(in module src.software.axon.packageInterface\)](#)
[\(in module src.software.container.basicTypes\)](#)
[\(in module src.software.container.indirection\)](#)
[\(in module src.software.dAMP.gatherMeta\)](#)
[\(in module src.software.dAMP.reportGenerator\)](#)
[\(in module src.software.datacontrol.dataControlGenMain\)](#)
[\(in module src.software.estimationROI\)](#)
[\(in module src.software.guiDeveloper\)](#)
[\(in module src.software.guiLayouts\)](#)
[\(in module src.software.guiTests\)](#)
[\(in module src.software.JIRA.analysisGuide\)](#)
[\(in module src.software.JIRA.classGetJiraIssues\)](#)
[\(in module src.software.JIRA.executeJiraMining\)](#)
[\(in module src.software.JIRA.jiraEmbedding\)](#)
[\(in module src.software.JIRA.jiraSimAnalysis\)](#)
[\(in module src.software.MEP.loadAndProbeDrive\)](#)
[\(in module src.software.MEP.mediaErrorPredictor\)](#)
[\(in module src.software.parse.pacmanIC\)](#)

(in module src.software.threadModuleAPI)	src.softw
(in module src.software.TSV.configToText)	src.softw
(in module src.software.TSV.DefragHistoryGrapher)	src.softw
(in module src.software.TSV.formatTSFiles)	src.softw
(in module src.software.TSV.generateTSBinaries)	src.softw
(in module src.software.TSV.loadAndProbeSystem)	src.softw
(in module src.software.TSV.visualizeTS)	src.softw
(in module src.software.userConfigurationProfile)	src.softw
(in module src.software.utilsCommon)	src.softw
(src.software.guiLayouts.dataTablePopulate method)	src.softw
main_estimator() (in module src.software.estimationROI)	src.softw
mainloop() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	src.softw
(src.software.MEP.mediaErrorGUI.StartPage method)	src.softw
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	src.softw
(src.software.TSV.genericObjectGUI.StartPage method)	src.softw
majorVersion	src.softw
(src.software.parse.intelTelemetryDataObject.intelTelemetryDataObjectHeaderV2_0_struct attribute)	src.softw
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V2_struct attribute)	src.softw
(src.software.parse.intelVUTelemetry.intelTelemetryTOC_V1_struct attribute)	src.softw
(src.software.parse.intelVUTelemetry.intelTelemetryTOC_V2_struct attribute)	src.softw
(src.software.parse.intelVUTelemetry.intelTelemetryTOC_VersionStruct attribute)	src.softw
(src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct attribute)	src.softw
make_table() (src.software.guiDeveloper.GUIDeveloper method)	src.softw
(src.software.guiLayouts.dataTablePopulate method)	src.softw
makeDataset() (src.software.autoAI.mediaPredictionRNN.WindowGenerator method)	src.softw
makeFoldersByFileType() (in module src.software.utilsCommon)	src.softw
manage() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	src.softw
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	src.softw
markFirstEntry()	src.softw
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventTupleTranslate method)	src.softw
MassiveParallelismmanyFunctionManyParameters (class in src.software.threadModuleAPI)	src.softw
MassiveParallelismSingleFunctionManyParameters (class in src.software.threadModuleAPI)	src.softw
matchFiles() (in module src.software.utilsCommon)	src.softw
matchWithCSV() (src.software.datacontrol.dataControlGenMain.DatacontrolGenH method)	src.softw
MAX_PARAMETER_COUNT	src.softw
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventHeader_union attribute)	src.softw
maxDiff (src.software.axon.axonInterface.Axon_Interface.TestAxonProducer attribute)	src.softw
(src.software.guiTests.AxonRegressionTests attribute)	src.softw
(src.software.guiTests.TableRegressionTests attribute)	src.softw
maxName (src.software.utility.templateUtility.templateUtility attribute)	src.softw
maxPath (src.software.utility.templateUtility.templateUtility attribute)	src.softw
maxsize() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	src.softw
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	src.softw
maxTime (src.software.utility.templateUtility.templateUtility attribute)	src.softw
MediaErrorPredictor (class in src.software.MEP.mediaErrorPredictor)	src.softw
mediaId (src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V2_struct attribute)	src.softw
mediaPredictionRNN (class in src.software.autoAI.mediaPredictionRNN)	src.softw
meta() (src.software.axon.axonMeta.AXON_Meta static method)	src.softw

milestoneList (src.software.JIRA.classSupportFiles.support_file attribute)	src.softw
minorVersion	moveFile() (i
(src.software.parse.intelTelemetryDataObject.intelTelemetryDataObjectHeaderV2_0_struct attribute)	MultiThreadF
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V2_struct attribute)	
(src.software.parse.intelVUTelemetry.intelTelemetryTOC_V1_struct attribute)	
(src.software.parse.intelVUTelemetry.intelTelemetryTOC_V2_struct attribute)	
(src.software.parse.intelVUTelemetry.intelTelemetryTOC_VersionStruct attribute)	
(src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct attribute)	
minsize() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	
ML_COMMENT_TOKEN (src.software.parse.parserSrcUtil.fileTokenizer attribute)	
(src.software.parse.parserSrcUtil.tokenList attribute)	
(src.software.parse.structdefParser.structdefTokenizer attribute)	

N

name (src.software.parse.bufdata.BufData attribute)	NlogParamete
(src.software.parse.bufdata.BufDataList attribute)	nlogPauseStat
(src.software.parse.bufdata.DirectAccess attribute)	(src.software.r
nameSeparator (src.software.parse.autoObjects.structDef attribute)	attribute)
(src.software.parse.parserDictionaryGen.parserDictionaryGenerator attribute)	(src.softw
(src.software.parse.parserTwidlGen.parserTwidlGenerator attribute)	attribute)
(src.software.parse.parserXmlGen.parserXmlGenerator attribute)	NLogPoolArr
nameToUidConverterGen() (src.software.parse.internal.twidlDictGen.twidlDictGenerator method)	src.software.p
nametowidget() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	NlogPredictor
(src.software.MEP.mediaErrorGUI.StartPage method)	nlogPredictor/
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	nlogPrimaryBi
(src.software.TSV.genericObjectGUI.StartPage method)	(src.software.p
NASA_Data (class in src.software.estimationROI)	attribute)
nearest_neighbor() (src.software.dAMP.iSOM.iSOM static method)	NlogSelect_sti
neuralNetClassify() (src.software.guiLayouts.GUILayouts static method)	src.software.p
newUser() (src.software.userConfigurationProfile.userConfigurationProfile method)	NlogSelect_ur
newUserGUI() (src.software.userConfigurationProfile.userConfigurationProfile method)	src.software.p
next() (src.software.mp.order.linearOrder method)	NlogSelectV2
(src.software.mp.order.Order method)	src.software.p
(src.software.mp.order.randomOrder method)	NlogSelectV4
nlogBufNum	src.software.p
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV4_struct attribute)	NlogSelectVer
nlogBufNumMax	src.software.p
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV4_struct attribute)	nlogTable (src
nlogByteSize	NlogTimePred
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV2_struct attribute)	NlogTimePred
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV4_struct attribute)	NlogTokenizer
NlogDataProcessor (class in src.software.autoAI.nlogPrediction)	nlogTriageBas
nlogEnumTranslate (class in src.software.parse.nlogParser.telemetry_parsers.nlogEnum)	NlogUtils (cl
	NlogWidthPre
	normalize_dat
	NPSG_Data (c
	NPSG_Progra
	number() (src

NlogEventPoolParser (class in src.software.parse.nlogParser.telemetry.parsers.telemetry_nlog)	(src.softw
NlogEventsPredictor (class in src.software.autoAI.nlogPrediction)	NUMERIC_T
NlogEventsPredictor.RNNUtility (class in src.software.autoAI.nlogPrediction)	(src.softw
nlogFolder (src.software.dAMP.gatherMeta .GatherMeta attribute)	numericParser
nlogFormatError() (src.software.parse.nlogParser.test_util.output_log .OutputLog static method)	NumParams
nlogId	(src.software.p
(src.software.parse.nlogParser.telemetry.parsers.telemetry_nlog.NlogSelectV2_struct attribute)	attribute)
(src.software.parse.nlogParser.telemetry.parsers.telemetry_nlog.NlogSelectV4_struct attribute)	NvmeControll
nlogName	NvmeControll
(src.software.parse.nlogParser.telemetry.parsers.telemetry_nlog.NlogSelectV2_struct attribute)	src.software.p
(src.software.parse.nlogParser.telemetry.parsers.telemetry_nlog.NlogSelectV4_struct attribute)	NvmeCtrlIniti
NlogParameterPredictor (class in src.software.autoAI.nlogPrediction)	attribute)
	NvmeHostInit
	NvmeHostInit
	NvmeHostInit
	NvmeLogId (s
	attribute)

O

OBJECT_TYPE (src.software.parse.autoObjects.structDef attribute)	OneShotExecuteAP
ObjectConfig (class in src.software.MEP.mediaErrorGUI)	OneShotLayout() (s
(class in src.software.TSV.genericObjectGUI)	OneShotTaskConfig
ObjectConfigARMA (class in src.software.guiCommon)	open() (src.software
ObjectConfigGenericObject (class in src.software.guiCommon)	OpenFiles (class in : src.software
ObjectConfigRNN (class in src.software.guiCommon)	openHtmlOutput() (src.software
objectDefine (class in src.software.parse.autoObjects)	OPERATOR_TOKI
objectEnum (src.software.parse.intelVUTelemetry.intelLogObjectInformation_struct attribute)	(src.software.p
objectEUID	(src.software.p
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V2_struct attribute)	(src.software.p
objectFilePath (src.software.guiCommon.ObjectConfigARMA attribute)	(src.software.p
(src.software.guiCommon.ObjectConfigGenericObject attribute)	(src.software.N
(src.software.guiCommon.ObjectConfigRNN attribute)	(src.software.T
(src.software.MEP.mediaErrorGUI.ObjectConfig attribute)	(src.software.T
(src.software.TSV.genericObjectGUI.ObjectConfig attribute)	OperatorList (src.so
objectHeaderSize	option_add() (src.so
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_4_struct attribute)	method)
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_struct attribute)	(src.software.N
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V2_struct attribute)	(src.software.T
objectIDDecode (src.software.guiCommon.ObjectConfigGenericObject attribute)	(src.software.T
objectIdentifier	(src.software.T
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_4_struct attribute)	option_get() (src.so
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_struct attribute)	method)
objectIDs (src.software.guiCommon.ObjectConfigARMA attribute)	(src.software.N
(src.software.guiCommon.ObjectConfigGenericObject attribute)	(src.software.T
(src.software.guiCommon.ObjectConfigRNN attribute)	(src.software.T
(src.software.MEP.mediaErrorGUI.ObjectConfig attribute)	Order (class in src.s

```

    (src.software.TSV.genericObjectGUI.ObjectConfig attribute)
objectList (src.software.parse.structdefParser.structdefParser attribute)
objects (src.software.parse.intelVUTelemetry.intelTelemetryTOC_V1_struct
attribute)
    (src.software.parse.intelVUTelemetry.intelTelemetryTOC_V2_struct attribute)
objectSizeBlks
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_4_struct
attribute)
    (src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_struct
attribute)
objectsOfInterest (src.software.dAMP.gatherMeta.GatherMeta attribute).
objectTextIdentifier
(src.software.parse.intelVUTelemetry.intelLogObjectInformation_struct attribute)
    (src.software.parse.intelVUTelemetry.intelLogObjectInformationV_1_4_struct
attribute)
    (src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_4_struct
attribute)
    (src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V1_struct
attribute)
objectTimeSeriesVisualizer() (src.software.guiLayouts.GUILayouts static method)
objectToDictionary() (in module src.software.JIRA.analysisGuide)
    (src.software.DP.preprocessingAPI.preprocessingAPI static method)
obtainEventSignature() (src.software.autoAI.nlogPrediction.NlogTokenizer method)
OctalDigitList (src.software.parse.parserSrcUtil.numericParser attribute)
    (src.software.parse.parserSrcUtil.parserHelper attribute)
    (src.software.parse.structdefParser.structdefParser attribute)
offsetBytes
(src.software.parse.intelVUTelemetry.telemetryTOCObjectEntryV2_struct attribute)
OneAPI (class in src.software.container.indirection)
OneAPI.Indirection (class in src.software.container.indirection)
OneAPI.Indirection.Lookup (class in src.software.container.indirection)
OneAPI.Indirection.Reflect (class in src.software.container.indirection)

```

P

pack() (src.software.MEP.mediaErrorGUI.StartPage method)	placeholder
(src.software.TSV.genericObjectGUI.StartPage method)	platform
pack_configure() (src.software.MEP.mediaErrorGUI.StartPage method)	plot().(s
(src.software.TSV.genericObjectGUI.StartPage method)	plot_ts
pack_forget() (src.software.MEP.mediaErrorGUI.StartPage method)	plotClu
(src.software.TSV.genericObjectGUI.StartPage method)	method
pack_info() (src.software.MEP.mediaErrorGUI.StartPage method)	plotEm
(src.software.TSV.genericObjectGUI.StartPage method)	method
pack_propagate() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	plotPer
(src.software.MEP.mediaErrorGUI.StartPage method)	(src.sof
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	populat
(src.software.TSV.genericObjectGUI.StartPage method)	(sr
pack_slaves() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	populat
(src.software.MEP.mediaErrorGUI.StartPage method)	(sr
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	populat
(src.software.TSV.genericObjectGUI.StartPage method)	method
packageInterface (class in src.software.axon.packageInterface)	populat
PACKED_DATA	populat
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_TelemetryV2NlogEventParserL2 attribute)	(src.sof
	static n

pacManIC (class in src.software.parse.pacmanIC)
pacmanICAPI().(src.software.parse.pacmanIC.pacManIC method)
pad_default().(src.software.axon.packageInterface.Cipher_AES method)
pad_pkcs5().(src.software.axon.packageInterface.Cipher_AES method)
pad_user_defined().(src.software.axon.packageInterface.Cipher_AES method)
paddEncodingList().(src.software.autoAI.nlogPrediction.NlogDataProcessor static method)
paddParamList().(src.software.autoAI.nlogPrediction.NlogDataProcessor static method)
paddParams().(src.software.autoAI.nlogPrediction.NlogDataProcessor static method)
paddTimeList().(src.software.autoAI.nlogPrediction.NlogDataProcessor static method)
padrActionStartMarker (src.software.parse.nlogParser.nlog_triage.padrTriage.padrTriage
attribute)
padrActionStartMarkerLen (src.software.parse.nlogParser.nlog_triage.padrTriage.padrTriage
attribute)
padrMarker (src.software.parse.nlogParser.nlog_triage.padrTriage.padrTriage attribute)
padrMarkerLen (src.software.parse.nlogParser.nlog_triage.padrTriage.padrTriage attribute)
padrStateEndMarker (src.software.parse.nlogParser.nlog_triage.padrTriage.padrTriage attribute)
padrTriage (class in src.software.parse.nlogParser.nlog_triage.padrTriage)
pageLogin().(src.software.JIRA.analysisGuide.AnalysisGuide method)
paramPredictorAPI().(src.software.autoAI.nlogPrediction.NlogPredictor method)
parse().(src.software.parse.headerTelemetry.TelemetryLogPageHeader_union method)
 (src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelect_union method)
 (src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.UnionBase method)
parseBinaryOnly().(src.software.TSV.generateTSBinaries.generateTSBinaries method)
parseData().(src.software.parse.nlogParser.telemetry_parsers.Level2Parser.Level2Parser method)
parseDefFile().(src.software.parse.structdefParser.structdefParser method)
parseError().(src.software.parse.parserSrcUtil.parserHelper method)
 (src.software.parse.structdefParser.structdefParser method)
parseInformation().(src.software.parse.parserSrcUtil.parserHelper method)
 (src.software.parse.structdefParser.structdefParser method)
parseInputs().(in module src.software.MEP.loadAndProbeDrive)
 (in module src.software.TSV.loadAndProbeSystem)
parseJson().(src.software.JIRA.classJiraIssues.jira_issue method)
ParseNlogData().(src.software.parse.intelObjectIdList.ParserObjectMap method)
ParseObjectData().(src.software.parse.intelObjectIdList.ParserObjectMap method)
parserDictionaryGenerator (class in src.software.parse.parserDictionaryGen)
parserHelper (class in src.software.parse.parserSrcUtil)
ParserObjectMap (class in src.software.parse.intelObjectIdList)
parserTwidlGenerator (class in src.software.parse.parserTwidlGen)
parserXmlGenerator (class in src.software.parse.parserXmlGen)
parseStream().(src.software.parse.parserSrcUtil.fileTokenizer method)
 (src.software.parse.structdefParser.structdefTokenizer method)
parseTelemetry().(src.software.parse.telemetryCmd.TelemetryObjectCommands method)
parseWarning().(src.software.parse.parserSrcUtil.parserHelper method)
 (src.software.parse.structdefParser.structdefParser method)
partitionDataFrame().(src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method)
partitionDriveWindows().(src.software.TSV.loadAndProbeSystem.loadSystem method)
pendingDocument (src.software.dAMP.reportGenerator.ReportGenerator attribute)
performSleep().(src.software.utilsCommon.FunctionScheduleTimer method)
performUserInputAPI().(src.software.JIRA.analysisGuide.AnalysisGuide method)
pickNCustersGMM().(src.software.JIRA.jiraSimAnalysis.jiraSimAnalysis method)
place().(src.software.MEP.mediaErrorGUI.StartPage method)
 (src.software.TSV.genericObjectGUI.StartPage method)
place_configure().(src.software.MEP.mediaErrorGUI.StartPage method)
 (src.software.TSV.genericObjectGUI.StartPage method)
place_forget().(src.software.MEP.mediaErrorGUI.StartPage method)
 (src.software.TSV.genericObjectGUI.StartPage method)

place_info() (src.software.MEP.mediaErrorGUI.StartPage method)	product
(src.software.TSV.genericObjectGUI.StartPage method)	profile/
place_slaves() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	profileI
(src.software.MEP.mediaErrorGUI.StartPage method)	method
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	profileI
(src.software.TSV.genericObjectGUI.StartPage method)	program
	prompt
	(in
	(propag
	(method
	(sr
	(sr
	(sr
	(proto
	(method
	(sr
	(proto
	(pssDeb
	(src.soft
	(putTok
	(sr
	(python_

Q

queryJira() (in module src.software.JIRA.classGetJiraIssues)	quit() (src.software.MEP.mediaErrorGUI.G
queryWebpage()	method)
(src.software.JIRA.analysisGuide.AnalysisGuide method)	(src.software.MEP.mediaErrorGUI.Sta
queueAsynchronousEventRequest()	(src.software.TSV.genericObjectGUI.C
(src.software.parse.telemetry_drive.logDevice method)	method)
quiet	(src.software.TSV.genericObjectGUI.S
(src.software.parse.nlogParser.test_util.output_log.OutputLog	
attribute)	
(src.software.parse.output_log.OutputLog attribute)	

R

RAAD() (src.software.axon.axonMeta.AXON_Meta static method)	Reserved
RADString() (src.software.axon.axonMeta.AXON_Meta method)	(src.soft
randomOrder (class in src.software.mp.order)	attribute)
RangeMax (src.software.dAMP.iSOM.iSOM attribute)	(src
RawHtlmOutput() (src.software.JIRA.classHtmlGen.htmlGen method)	attri
read_window() (src.software.guiDeveloper.GUIDeveloper method)	(src
readConfigContent() (src.software.guiCommon.DefragConfig method)	attri
(src.software.guiCommon.ObjectConfigARMA method)	reserved
(src.software.guiCommon.ObjectConfigGenericObject method)	(src.soft
(src.software.guiCommon.ObjectConfigRNN method)	attribute)
(src.software.MEP.mediaErrorGUI.ObjectConfig method)	reserved
(src.software.TSV.genericObjectGUI.ObjectConfig method)	attribute)
readData() (src.software.JIRA.jiraSimAnalysis.jiraSimAnalysis method)	(src
readFile() (src.software.datacontrol.dataControlGenMain.DatacontrolGenCSV method)	attri

(src.software.datacontrol.dataControlGenMain.DatacontrolGenH method)	reserved
readFileBytes()(src.software.axon.axonInterface.Axon_Interface method)	(src.soft
readFileContentIntoDict()(src.software.autoAI.nlogPrediction.NlogDataProcessor static method)	attribute)
readFileIntoConfig()(src.software.DP.preprocessingAPI.preprocessingAPI static method)	(src
readFiles()(in module src.software.utilsCommon)	attri
readprofile()(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	(src
 (src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	attri
readProfile()(src.software.userConfigurationProfile.userConfigurationProfile method)	(src
readProfileAsDictionary()(src.software.userConfigurationProfile.userConfigurationProfile method)	attri
reason(src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct attribute)	reset()
reasonCode(src.software.parse.intelVUTelemetryReason.intelTelemetryReasonCode attribute)	(s
reasonId(src.software.parse.headerTelemetry.NvmeControllerInitiatedLogPageHeader_struct attribute)	resetPars
 (src	resetRec
 (src	resetStru
 (src	resetTok
 (src	resetTok
 (src	(src
 (src	reshape3
reattachDrive()(src.software.parse.internal.drive_utility.driveList static method)	resizable
receiveCmd()(src.software.axon.axonInterface.Axon_Interface method)	(src
record()(src.software.axon.axonMeta.AXON_Meta method)	restoreM
recoverAssertedDrive()(src.software.parse.telemetry_drive.logDevice method)	method)
reduceDimensionalityOfEmbeddings()(src.software.autoAI.nlogPrediction.NlogTokenizer method)	resultsFc
reformatString()(src.software.autoAI.nlogPrediction.NlogTokenizer method)	retrieveL
refreshMetaTable()(src.software.guiDeveloper.GUIDeveloper method)	retrieveL
register()(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	returnLa
 (src.software.MEP.mediaErrorGUI.StartPage method)	reverseD
 (src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	ridgeLin
 (src.software.TSV.genericObjectGUI.StartPage method)	RNNAc
RegisterTriageList()	RNNFig
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventTupleTranslate method)	RNNMo
RegisterTriageObject()	RNNMo
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventTupleTranslate method)	RNNPre
RegressionUnitTestsCases (class in src.software.dAMP.reportGenerator)	roundup]
remove_readonly()(in module src.software.datacontrol.dataControlGenMain)	(src
removeDuplicateStructures()(src.software.parse.autoObjects.structDefList method)	rowconfi
removeNullFields()(src.software.DP.preprocessingAPI.preprocessingAPI method)	method)
 (src	(src
 (src	(src
replaceFormattingForRegex()(src.software.autoAI.nlogPrediction.NlogTokenizer method)	(src
report_callback_exception()	Rows(si
(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	run()(sr
 (src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	(src
reportDictionary()(src.software.guiOneShot.GUIOneShot attribute)	(src
reportFlatDictionary()(src.software.guiOneShot.GUIOneShot attribute)	(src
ReportGenerator (class in src.software.dAMP.reportGenerator)	runCmd(
reportGenObj()(src.software.dAMP.reportGenerator.RegressionUnitTestsCases attribute)	runCtype
reportVersion(src.software.dAMP.reportGenerator.ReportGenerator attribute)	method)
reserved(src.software.parse.headerTelemetry.NvmeControllerInitiatedLogPageHeader_struct attribute)	runPhase
 (src.software.parse.headerTelemetry.NvmeHostInitiatedLogPageHeader_struct attribute)	
 (src.software.parse.headerTelemetry.SataControllerInitiatedLogPageHeader_struct attribute)	
 (src.software.parse.headerTelemetry.SataHostInitiatedLogPageHeader_struct attribute)	

```

(src.software.parse.intelTelemetryDataObject.intelTelemetryDataObjectHeaderV2_0_struct
attribute)
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader_V2_struct attribute)
(src.software.parse.intelVUTelemetry.intelTelemetryTOC_V1_struct attribute)
(src.software.parse.intelVUTelemetry.intelTelemetryTOC_V2_struct attribute)
(src.software.parse.intelVUTelemetryReason.intelTelemetryReasonCode attribute),

```

S

SataControllerInitiatedLogId (src.software.parse.telemetry_drive.logDevice attribute)	snippet
SataControllerInitiatedLogPageHeader_struct (class in src.software.parse.headerTelemetry)	source(
SataCtrlInitiated (src.software.parse.headerTelemetry.TelemetryLogPageHeader_union attribute)	SOURC
SataHostInitiated (src.software.parse.headerTelemetry.TelemetryLogPageHeader_union attribute)	(src.sof
SataHostInitiatedLogId (src.software.parse.telemetry_drive.logDevice attribute)	attribut
SataHostInitiatedLogPageHeader_struct (class in src.software.parse.headerTelemetry)	source(
SataLogId (src.software.parse.headerTelemetry.TelemetryLogPageHeader_union attribute)	Source(
saveIdentifyJSON() (src.software.axon.axonInterface.Axon_Interface method)	(src.sof
saveJSON() (src.software.axon.axonInterface.Axon_Interface method)	attribut
(src.software.axon.axonMeta.AXON_Meta method)	spawn^
savePath (src.software.dAMP.reportGenerator.ReportGenerator attribute)	specific
SaveProfile() (src.software.axon.axonProfile.AxonProfile method)	specific
ScanDrives() (in module src.software.parse.internal.drive_utility)	splitDa
scrumTeam (src.software.JIRA.classSupportFiles.support_file attribute)	splitWi
searchForSignature() (src.software.JIRA.analysisGuide.AnalysisGuide method)	method
searchForSignatureFromCode() (src.software.JIRA.analysisGuide.AnalysisGuide method)	sprintE
searchProfile() (src.software.userConfigurationProfile.userConfigurationProfile method)	src.soft
SECONDARY_VARS (src.software.TSV.genericObjectGUI.Fields attribute)	mc
secondaryVarLabels (src.software.guiCommon.GenericObjectGraph attribute)	src.soft
secondaryVars (src.software.guiCommon.DefragConfig attribute)	mc
seconds (src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventTimeStamp_struct attribute)	src.soft
secureHash() (src.software.axon.axonInterface.Axon_Interface method)	mc
SelBest() (src.software.JIRA.jiraSimAnalysis.jiraSimAnalysis static method)	src.soft
selectAddedOffset	mc
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_NlogSelectV4_struct attribute)	src.soft
selection_clear() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	src.soft
(src.software.MEP.mediaErrorGUI.StartPage method)	mc
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	src.soft
(src.software.TSV.genericObjectGUI.StartPage method)	mc
selection_get() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	src.soft
(src.software.MEP.mediaErrorGUI.StartPage method)	mc
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	src.soft
(src.software.TSV.genericObjectGUI.StartPage method)	mc
selection_handle() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	src.soft
(src.software.MEP.mediaErrorGUI.StartPage method)	mc
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	src.soft
(src.software.TSV.genericObjectGUI.StartPage method)	mc
selection_own() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	src.soft
(src.software.MEP.mediaErrorGUI.StartPage method)	mc
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	src.soft
(src.software.TSV.genericObjectGUI.StartPage method)	mc
selection_own_get() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	src.soft
(src.software.MEP.mediaErrorGUI.StartPage method)	mc
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	src.soft

(src.software.TSV.genericObjectGUI.StartPage method)	src.soft
selectNlogPause	src.soft
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV4_struct attribute)	src.soft
selectOffsetRef	src.soft
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.NlogSelectV4_struct attribute)	src.soft
send()_(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	src.soft
(src.software.MEP.mediaErrorGUI.StartPage method)	src.soft
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	src.soft
(src.software.TSV.genericObjectGUI.StartPage method)	src.soft
sendAPI()_(src.software.axon.axonInterface.Axon_Interface method)	src.soft
sendAXONDownloadCommand()_(src.software.guiDeveloper.GUIDeveloper method)	src.soft
sendCmd()_(src.software.axon.axonInterface.Axon_Interface method)	src.soft
sendInfo()_(src.software.axon.axonInterface.Axon_Interface method)	src.soft
sendRestRequest()_(src.software.JIRA.classGetJiraIssues.get_jira_issues method)	src.soft
sequentialNet()_(src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method)	src.soft
(src.software.autoAI.nlogPrediction.NlogEventsPredictor method)	src.soft
(src.software.autoAI.nlogPrediction.NlogParameterPredictor method)	src.soft
(src.software.autoAI.nlogPrediction.NlogTimePredictor method)	src.soft
serialNumber_(src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct attribute)	src.soft
set_analysisUIDs()_(src.software.JIRA.analysisGuide.HandbookMeta method)	src.soft
set_code()_(src.software.JIRA.analysisGuide.HandbookMeta method)	src.soft
set_direct_access()_(src.software.parse.bufdata.DirectAccess method)	src.soft
set_iv()_(src.software.axon.packageInterface.Cipher_AES method)	src.soft
set_key()_(src.software.axon.packageInterface.Cipher_AES method)	src.soft
set_knownCauses()_(src.software.JIRA.analysisGuide.HandbookMeta method)	src.soft
set_url()_(src.software.JIRA.analysisGuide.HandbookMeta method)	src.soft
setAll()_(src.software.estimationROI.NPSG_ProgramData method)	src.soft
setAppearance()_(src.software.guiDeveloper.GUIDeveloper method)	src.soft
setApplication()_(src.software.userConfigurationProfile.userConfigurationProfile method)	src.soft
setApplications()_(src.software.dAMP.gatherMeta.GatherMeta method)	src.soft
setARMAFigures()_(src.software.dAMP.gatherMeta.GatherMeta method)	src.soft
setAssistedFigures()_(src.software.dAMP.gatherMeta.GatherMeta method)	src.soft
setAssistedFiguresParallel()_(src.software.dAMP.gatherMeta.GatherMeta method)	src.soft
setBinDirectory()_(src.software.parse.pacmanIC.pacManIC method)	src.soft
setBlock0Size()_(src.software.parse.telemetry_drive.logDevice method)	src.soft
setBusinessUnit()_(src.software.container.basicTypes.UID_Org method)	src.soft
setCategoricalEncoding()_(src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method)	src.soft
setCiLog()_(src.software.parse.telemetry_drive.logDevice method)	src.soft
setCiNotification()_(src.software.parse.telemetry_drive.logDevice method)	src.soft
setData()_(src.software.container.basicTypes.UID_Application method)	src.soft
(src.software.container.basicTypes.UID_Data method)	src.soft
setDatakey()_(src.software.JIRA.classGetJiraIssues.get_jira_issues method)	src.soft
setDataLakeMeta()_(src.software.dAMP.gatherMeta.GatherMeta method)	src.soft
setDataSet()_(src.software.JIRA.jiraSimAnalysis.jiraSimAnalysis method)	src.soft
setDebug()_(src.software.dAMP.reportGenerator.ReportGenerator method)	src.soft
(src.software.TSV.configToText.configToText method)	src.soft
(src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher method)	src.soft
(src.software.TSV.formatTSFiles.formatTSFiles method)	src.soft
(src.software.TSV.generateTSBinaries.generateTSBinaries method)	src.soft
(src.software.TSV.visualizeTS.visualizeTS method)	src.soft
setDebugLevel()_(src.software.parse.nlogParser.test_util.output_log.OutputLog static method)	src.soft
(src.software.parse.output_log.OutputLog static method)	src.soft
setDescription()_(src.software.parse.autoObjects.structDef method)	src.soft
setDeviceConfiguration()_(src.software.dAMP.gatherMeta.GatherMeta method)	src.soft
setDeviceSignature()_(src.software.dAMP.gatherMeta.GatherMeta method)	src.soft

setEmbeddedEncoding()	(src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method)	src.soft
setFirmwareDFA()	(src.software.dAMP.gatherMeta.GatherMeta method)	mc
setFunctionName()		src.soft
(src.software.threadModuleAPI.MassiveParallelismSingleFunctionManyParameters method)		mc
setGeometry()	(src.software.dAMP.reportGenerator.ReportGenerator method)	src.soft
setGroup()	(src.software.container.basicTypes.UID_Org method)	mc
setHiLog()	(src.software.parse.telemetry_drive.logDevice method)	src.soft
setInitialize()	(src.software.dAMP.reportGenerator.ReportGenerator method)	mc
setIterations()	(src.software.TSV.generateTSBinaries.generateTSBinaries method)	src.soft
setJIRA()	(src.software.dAMP.gatherMeta.GatherMeta method)	mc
setLocalTimeZone()		src.soft
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_NlogEventPoolParser method)		mc
setMajor()	(src.software.container.basicTypes.UID_Application method)	src.soft
	(src.software.container.basicTypes.UID_Data method)	mc
	(src.software.container.basicTypes.VersionTracking method)	src.soft
setMajorVersion()	(src.software.parse.autoObjects.objectDefine method)	mc
	(src.software.parse.autoObjects.structDef method)	src.soft
setMatrixProfile()	(src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method)	mc
setMatrixProfileFlag()	(src.software.MEP.mediaErrorPredictor.MediaErrorPredictor method)	src.soft
setMinor()	(src.software.container.basicTypes.UID_Application method)	mc
	(src.software.container.basicTypes.UID_Data method)	src.soft
	(src.software.container.basicTypes.VersionTracking method)	mc
setMinorVersion()	(src.software.parse.autoObjects.objectDefine method)	src.soft
	(src.software.parse.autoObjects.structDef method)	mc
setMLClustering()	(src.software.dAMP.gatherMeta.GatherMeta method)	src.soft
setMLMinedJira()	(src.software.dAMP.gatherMeta.GatherMeta method)	mc
setMLProfiles()	(src.software.dAMP.gatherMeta.GatherMeta method)	src.soft
setName()	(src.software.parse.autoObjects.structDef method)	mc
setNlogBinFileName()		src.soft
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_TelemetryV2NlogEventParserL2 method)		mc
setNlogInfo()	(src.software.dAMP.gatherMeta.GatherMeta method)	src.soft
setNoInline()	(src.software.parse.autoObjects.structDef method)	src.soft
setObjStruct()	(src.software.parse.autoObjects.objectDefine method)	mc
setOutpath()	(src.software.TSV.formatTSFiles.formatTSFiles method)	src.soft
	(src.software.TSV.generateTSBinaries.generateTSBinaries method)	mc
setOutputFilename()	(src.software.DP.preprocessingAPI.preprocessingAPI method)	src.soft
	(src.software.JIRA.classGetJiraIssues.get_jira_issues method)	mc
setPairScore()	(in module src.software.JIRA.jiraEmbedding)	src.soft
setParametersList()		mc
(src.software.threadModuleAPI.MassiveParallelismSingleFunctionManyParameters method)		src.soft
SetPaths()	(src.software.guiOneShot.GUIOneShot method)	mc
setPoints	(src.software.guiCommon.DefragConfig attribute)	src.soft
setProjDirectory()	(src.software.parse.pacmanIC.pacManIC method)	mc
setRNNFigures()	(src.software.dAMP.gatherMeta.GatherMeta method)	src.soft
setScrumTeam()	(src.software.JIRA.classJiraIssuesList.jira_issues_lists method)	mc
setSecondaryVars()	(src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher method)	src.soft
setSetPointNames()	(src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher method)	mc
setSize()	(src.software.parse.autoObjects.objectDefine method)	src.soft
setSprintStartDate()	(src.software.JIRA.classJiraIssuesList.jira_issues_lists method)	mc
setSprintStartDateFromEnd()	(src.software.JIRA.classJiraIssuesList.jira_issues_lists method)	src.soft
setTeam()	(src.software.container.basicTypes.UID_Org method)	mc
setTelemetryMetaStats()	(src.software.dAMP.gatherMeta.GatherMeta method)	src.soft
 setTimeSeries()	(src.software.dAMP.gatherMeta.GatherMeta method)	mc
setTrackingNames()	(src.software.TSV.DefragHistoryGrapher.DefragHistoryGrapher method)	src.soft
setType()	(src.software.container.basicTypes.VersionTracking method)	mc

<u>setUID()</u> (src.software.container.basicTypes.UID_Application method)	src.soft
(src.software.container.basicTypes.UID_Data method)	mc
(src.software.container.basicTypes.VersionTracking method)	src.soft
<u>setUid()</u> (src.software.parse.autoObjects.objectDefine method)	src.soft
(src.software.parse.autoObjects.structDef method)	mc
<u>SetUlink()</u> (in module src.software.parse.internal.drive_utility).	src.soft
<u>setup()</u> (in module src.software.axon.axonInterface)	src.soft
(in module src.software.axon.axonMeta)	mc
(in module src.software.axon.packageInterface)	src.soft
<u>setUp()</u> (src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method)	src.soft
(src.software.dAMP.reportGenerator.RegressionUnitTestsCases method)	mc
(src.software.guiTests.AxonRegressionTests method)	starting
(src.software.guiTests.TableRegressionTests method)	(src.sof
<u>setup()</u> (src.software.utilsCommon.FunctionScheduleTimer method)	attribut
<u>setUpClass()</u> (src.software.axon.axonInterface.Axon_Interface.TestAxonProducer class method)	StartPa
(src.software.guiTests.AxonRegressionTests class method)	(cl
(src.software.guiTests.TableRegressionTests class method)	state()_(
<u>setUser()</u> (src.software.JIRA.analysisGuide.AnalysisGuide method)	(sr
(src.software.userConfigurationProfile.userConfigurationProfile method)	stateDic
<u>setUsernamePassword()</u> (src.software.JIRA.analysisGuide.AnalysisGuide method)	static n
<u>setValue()</u> (src.software.guiCommon.dictElement method)	status(:
<u>setvar()</u> (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	StepsM
(src.software.MEP.mediaErrorGUI.StartPage method)	stretchC
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	STRIN
(src.software.TSV.genericObjectGUI.StartPage method)	(sr
<u>setVendor()</u> (src.software.container.basicTypes.UID_Org method)	(sr
<u>setWarnIsError()</u> (src.software.parse.nlogParser.test_util.output_log.OutputLog static method)	(sr
(src.software.parse.output_log.OutputLog static method)	strip_ei
<u>shortDescription()</u> (src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method)	strip_st
(src.software.guiTests.AxonRegressionTests method)	strip_Si
(src.software.guiTests.TableRegressionTests method)	stripCo
<u>show_frame()</u> (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	STRUCT
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	structD
<u>sightingsBCH()</u> (src.software.JIRA.classCommonReports.common_reports method)	structD
<u>sightingsSummaryByProgram()</u> (src.software.JIRA.classCommonReports.common_reports method)	structde
<u>silent</u> (src.software.parse.nlogParser.test_util.output_log.OutputLog attribute)	structde
<u>similar()</u> (src.software.parse.bufdata.BufData method)	structLi
(src.software.parse.bufdata.BufDataList method)	subdirN
(src.software.parse.bufdata.DirectAccess method)	attribut
<u>SimilarityScoreCalculate()</u> (in module src.software.utilsCommon)	(sr
<u>singularVectorDecomp()</u> (src.software.JIRA.jiraSimAnalysis.jiraSimAnalysis method)	(sr
<u>size()</u> (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	subTest
(src.software.MEP.mediaErrorGUI.StartPage method)	method
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	(sr
(src.software.TSV.genericObjectGUI.StartPage method)	(sr
<u>sizefrom()</u> (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	suite()
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)	(sr
<u>sizeInDec()</u> (in module src.software.datacontrol.dataControlGenMain)	support
<u>skipTest()</u> (src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method)	support
(src.software.guiTests.AxonRegressionTests method)	attribut
(src.software.guiTests.TableRegressionTests method)	system(
<u>SL_COMMENT_TOKEN</u> (src.software.parse.parserSrcUtil.fileTokenizer attribute)	system)
(src.software.parse.parserSrcUtil.tokenList attribute)	
(src.software.parse.structdefParser.structdefTokenizer attribute)	
<u>slaves()</u> (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)	

[\(src.software.MEP.mediaErrorGUI.StartPage method\)](#)
[\(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method\)](#)
[\(src.software.TSV.genericObjectGUI.StartPage method\)](#).

T

[tabAPI\(\)](#)
[\(src.software.guiLayouts.GUILayouts method\)](#)
[tabDelimitedToList\(\)](#)
[\(src.software.JIRA.classSupportFiles.support_file method\)](#)
[TableData\(\)](#)
[\(src.software.JIRA.classHtmlGen.htmlGen method\)](#)
[TableDataColor\(\)](#)
[\(src.software.JIRA.classHtmlGen.htmlGen method\)](#)
[TableHeader\(\)](#)
[\(src.software.JIRA.classHtmlGen.htmlGen method\)](#)
[TableRegressionTests\(class in src.software.guiTests\)](#)
[teamIssueList](#)
[\(src.software.JIRA.classJiraIssuesList.jira_issues_lists attribute\)](#)
[teamName](#)
[\(src.software.dAMP.reportGenerator.ReportGenerator attribute\)](#)
[tearDown\(\)](#)
[\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method\)](#)
[\(src.software.dAMP.reportGenerator.RegressionUnitTestsCases method\)](#)
[\(src.software.guiTests.AxonRegressionTests method\)](#)
[\(src.software.guiTests.TableRegressionTests method\)](#)
[tearDownClass\(\)](#)
[\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer class method\)](#)
[\(src.software.guiTests.AxonRegressionTests class method\)](#)
[\(src.software.guiTests.TableRegressionTests class method\)](#)
[telemetryBinary](#)
[\(src.software.container.indirection.OneAPI attribute\)](#)
[TelemetryData](#)
[\(class in src.software.container.basicTypes\)](#)
[telemetryHeaderDict](#)
[\(src.software.dAMP.gatherMeta.GatherMeta attribute\)](#)
[telemetryHostTimeV1_0](#)
[\(class in src.software.parse.nlogParser.telemetry.parsers.telemetryHostTimeV1_0\)](#)
[TelemetryLogPageHeader_union](#)
[\(class in src.software.parse.headerTelemetry\)](#)
[telemetryMetaStats](#)
[\(src.software.dAMP.gatherMeta.GatherMeta attribute\)](#)
[TelemetryObjectCommands](#)
[\(class in src.software.parse.telemetryCmd\)](#)
[telemetryStruct_union](#)
[\(class in src.software.parse.intelTelemetryDataObject\)](#)
[telemetryTOBJECTCatalogEntry_struct](#)
[\(class in src.software.parse.intelVUTelemetry\)](#)
[telemetryTOCOObjectEntryV2_struct](#)
[\(class in src.software.parse.intelVUTelemetry\)](#)
[TelemetryV2NlogEventParserL2](#)
[\(class in src.software.parse.nlogParser.telemetry.parsers.telemetry_nlog\)](#)
[telemetryVersion](#)
[\(src.software.container.indirection.OneAPI attribute\)](#)
[templateUtility](#)
[\(class in src.software.utility.templateUtility\)](#)
[test](#)
[\(src.software.autoAI.mediaPredictionRNN.WindowGenerator property\)](#)
[test\(\)](#)
[\(in module src.software.parse.internal.twidlDictGen\)](#)
[\(src.software.JIRA.classJiraIssuesList.jira_issues_lists method\)](#)
[test_addMathMatrix\(\)](#)
[\(src.software.dAMP.reportGenerator.RegressionUnitTestsCases method\)](#)
[test_addMetaTable\(\)](#)
[\(src.software.dAMP.reportGenerator.RegressionUnitTestsCases method\)](#)
[test_addPDFImages\(\)](#)
[\(src.software.dAMP.reportGenerator.RegressionUnitTestsCases method\)](#)
[test_create_large_file\(\)](#)
[\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method\)](#)
[test_createDocument\(\)](#)
[\(src.software.dAMP.reportGenerator.RegressionUnitTestsCases method\)](#)
[test_send_page_data\(\)](#)
[\(src.software.axon.axonInterface.Axon_Interface.TestAxonProducer method\)](#)
[test_updateCover\(\)](#)
[\(src.software.dAMP.reportGenerator.RegressionUnitTestsCases method\)](#)
[test_updateCustomerInformation\(\)](#)
[\(src.software.dAMP.reportGenerator.RegressionUnitTestsCases method\)](#)
[testCase\(\)](#)
[\(src.software.axon.packageInterface.Cipher_AES static method\)](#)
[testCorrectUploadID\(\)](#)
[\(src.software.guiTests.AxonRegressionTests method\)](#)
[testDownloadBadID\(\)](#)
[\(src.software.guiTests.AxonRegressionTests method\)](#)
[testDownloadSuccess\(\)](#)
[\(src.software.guiTests.AxonRegressionTests method\)](#)

[testDrive](#) ([class in src.software.parse.internal.drive_utility](#)).
[testEmptyUpload\(\)](#) ([src.software.guiTests.AxonRegressionTests method](#))
[testGetDataDict\(\)](#) ([src.software.guiTests.TableRegressionTests method](#))
[testHarness\(\)](#) ([src.software.axon.packageInterface.Cipher_AES method](#))
[testNilContainer\(\)](#) ([src.software.guiTests.TableRegressionTests method](#))
[testOneShotContainer\(\)](#) ([src.software.guiTests.TableRegressionTests method](#))
[testPopupTable\(\)](#) ([src.software.guiTests.TableRegressionTests method](#))
[testSimpleContainer\(\)](#) ([src.software.guiTests.TableRegressionTests method](#))
[testSize\(\)](#) ([in module src.software.parse.internal.twidlDictGen](#))
[testSpecificContainer\(\)](#) ([src.software.guiTests.TableRegressionTests method](#))
[testSuite\(\)](#) ([in module src.software.dAMP.gatherMeta](#))
 ([in module src.software.dAMP.reportGenerator](#))
[testUploadFail\(\)](#) ([src.software.guiTests.AxonRegressionTests method](#))
[testUploadSuccess\(\)](#) ([src.software.guiTests.AxonRegressionTests method](#))
[testUploadSuccessDev\(\)](#) ([src.software.guiTests.AxonRegressionTests method](#))
[text_verify\(\)](#) ([src.software.axon.packageInterface.Cipher_AES method](#))
[tfidfVectorization\(\)](#) ([src.software.JIRA.jiraEmbedding.jiraEmbedding method](#))
[threadLoop\(\)](#) ([src.software.threadModuleAPI.MultiThreadFL method](#))
[threadProcessLoop\(\)](#) ([src.software.threadModuleAPI.MultiThreadFL static method](#))
[threadSplitProcessing\(\)](#) ([src.software.threadModuleAPI.MultiThreadFL static method](#))
[threshold](#) ([src.software.JIRA.classSupportFiles.support_file attribute](#))
[ticket\(\)](#) ([src.software.axon.axonMeta.AXON_Meta static method](#))
[ticketString\(\)](#) ([src.software.axon.axonMeta.AXON_Meta method](#))
[ticks](#) ([src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventTimeStamp_struct attribute](#))
 ([ticksPerSecond](#)
 ([src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_NlogSelectV2_struct attribute](#))
 ([src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_NlogSelectV4_struct attribute](#))
[time0Marker](#)
 ([src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventTimeStamp_struct attribute](#))
[TIME_ADJUSTMENT_TOKEN](#)
 ([src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventHeader_union attribute](#))
[timePredictorAPI\(\)](#) ([src.software.autoAI.nlogPrediction.NlogPredictor method](#))
[timeSeriesPredictorAllAPI\(\)](#) ([src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method](#))
[timeSeriesPredictorAPI\(\)](#) ([src.software.autoAI.mediaPredictionRNN.timeSeriesRNN method](#))
[timeSeriesRNN](#) ([class in src.software.autoAI.mediaPredictionRNN](#))
[timeSeriesRNN.RNNUtility](#) ([class in src.software.autoAI.mediaPredictionRNN](#))
[timeSeriesSignatures](#) ([src.software.dAMP.gatherMeta.GatherMeta attribute](#))
[timeStamp](#) ([src.software.guiOneShot.GUIOneShot attribute](#))
 ([src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventTimeStamp_union attribute](#))
[timestamp](#)
 ([src.software.parse.nlogParser.telemetry_parsers.telemetryHostTimeV1_0.telemetryHostTimeV1_0 attribute](#))
[TimestampEnd](#) ([src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_NlogSelect_struct attribute](#))
[TimestampStart](#) ([src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_NlogSelect_struct attribute](#))
[timeToken](#) ([src.software.dAMP.reportGenerator.ReportGenerator attribute](#))
[title\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
[titleInfo](#) ([src.software.dAMP.reportGenerator.ReportGenerator attribute](#))
[tk_bisque\(\)](#) ([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))

([src.software.TSV.genericObjectGUI.StartPage](#) method)

U

[uEXE\(\)](#) ([src.software.utility.templateUtility.templateUtility](#) method)
[UID](#) (class in [src.software.datacontrol.dataControlGenMain](#))
[UID_Application](#) (class in [src.software.container.basicTypes](#))
[UID_Data](#) (class in [src.software.container.basicTypes](#))
[UID_Org](#) (class in [src.software.container.basicTypes](#))
[uidInitByList\(\)](#)
([src.software.datacontrol.dataControlGenMain](#).[DatacontrolGenCSV](#) method)
[uidObject](#) ([src.software.container.indirection](#).[OneAPI_Indirection](#) attribute)
[uidsFound](#) ([src.software.dAMP.gatherMeta](#).[GatherMeta](#) attribute)
[uidSort\(\)](#) (in module [src.software.datacontrol.dataControlGenMain](#))
[uidTempSort\(\)](#) (in module [src.software.datacontrol.dataControlGenMain](#))
[uImportDirectoryTree\(\)](#) ([src.software.utility.templateUtility](#).[templateUtility](#) method)
[uInit\(\)](#) ([src.software.utility.templateUtility](#).[templateUtility](#) method)
[uMain\(\)](#) ([src.software.utility.templateUtility](#).[templateUtility](#) method)
[unbind\(\)](#)
([src.software.MEP.mediaErrorGUI](#).[GenericObjectAppMainWindow](#) method)
 ([src.software.MEP.mediaErrorGUI](#).[StartPage](#) method)
 ([src.software.TSV.genericObjectGUI](#).[GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI](#).[StartPage](#) method)
[unbind_all\(\)](#)
([src.software.MEP.mediaErrorGUI](#).[GenericObjectAppMainWindow](#) method)
 ([src.software.MEP.mediaErrorGUI](#).[StartPage](#) method)
 ([src.software.TSV.genericObjectGUI](#).[StartPage](#) method)
[unbind_class\(\)](#)
([src.software.MEP.mediaErrorGUI](#).[GenericObjectAppMainWindow](#) method)
 ([src.software.MEP.mediaErrorGUI](#).[StartPage](#) method)
 ([src.software.TSV.genericObjectGUI](#).[GenericObjectAppMainWindow](#) method)
 ([src.software.TSV.genericObjectGUI](#).[StartPage](#) method)
[UNION_TYPE](#) ([src.software.parse.autoObjects](#).[structDef](#) attribute)
[UnionBase](#) (class in [src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog](#))
[uniqueIdentifierTypes_e\(\)](#) (in module [src.software.container.basicTypes](#))
[uniqueLists](#) ([src.software.autoAI](#).[transformCSV](#).[TransformMetaData](#) attribute)
[unitTest\(\)](#) (in module [src.software.JIRA.analysisGuide](#))
[UNKNOWN_TYPE](#) ([src.software.parse.autoObjects](#).[structDef](#) attribute)
[unlockDrive\(\)](#) ([src.software.parse_internal_drive_utility](#).[testDrive](#) method)
[unpad_default\(\)](#) ([src.software.axon.packageInterface](#).[Cipher_AES](#) method)
[unpad_pkcs5\(\)](#) ([src.software.axon.packageInterface](#).[Cipher_AES](#) method)
[unpad_user_defined\(\)](#) ([src.software.axon.packageInterface](#).[Cipher_AES](#) method)

[update_idletasks\(\)](#) ([src.softwa](#) method)
 ([src.software.MEP.media](#)
 ([src.software.TSV.generi](#)
 ([src.software.TSV.generi](#)
[updateCover\(\)](#) ([src.software.d](#)
[updateCustomerInformation\(\)](#)
 method)
[updateFile\(\)](#) ([src.software.dat](#)
 ([src.software.datacontrol](#)
[updateInitialize\(\)](#) ([src.softwar](#)
[updateOffset\(\)](#) ([src.software.p](#)
[updatePythonDict\(\)](#) ([src.softw](#)
 method)
[updateReportMeta\(\)](#) ([src.softy](#)
[updateSnapshot\(\)](#) ([src.softwar](#)
[updateState\(\)](#) ([src.software.au](#)
[updateStuctData\(\)](#) ([src.softwa](#)
[updateTime\(\)](#)
([src.software.parse.nlogParse](#)
 method)
 ([src.software.parse.nlogF](#)
 ([src.software.parse.nlogF](#)
 method)
[updateTimeStamp\(\)](#) ([src.softw](#)
 ([src.software.guiDevelop](#)
[updateToBitField\(\)](#) ([src.softw](#)
 ([src.software.parse.autoC](#)
[upload\(\)](#) ([src.software.guiLay](#)
[urlRoot](#) ([src.software.JIRA.cl](#)
 ([src.software.JIRA.class](#)
 ([src.software.JIRA.class](#)
[userConfigurationProfile](#) (clas
[userConfigurationProfile.appl](#)
[src.software.userConfiguratio](#)
[userConfigurationProfile.user](#)
[userFeedback\(\)](#) ([src.software.](#)
[userName](#) ([src.software.dAM](#)
[userProfileCreate\(\)](#) ([src.softw](#)
[userProfileLoad\(\)](#) ([src.softwa](#)
[utilityTSV](#) (class in [src.softw](#)

```

update\(\)
(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow
method)
(src.software.MEP.mediaErrorGUI.StartPage method)
(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow
method)
(src.software.TSV.genericObjectGUI.StartPage method).

```

V

```

v1Struct (src.software.parse.intelVUTelemetry.intelTelemetryTOC\_union attribute),
v2Struct (src.software.parse.intelVUTelemetry.intelTelemetryTOC\_union attribute),
val (src.software.autoAI.mediaPredictionRNN.WindowGenerator property),
validate\(\).
(src.software.parse.headerTelemetry.NvmeControllerInitiatedLogPageHeader\_struct
method)
(src.software.parse.headerTelemetry.NvmeHostInitiatedLogPageHeader\_struct
method)
(src.software.parse.headerTelemetry.SataControllerInitiatedLogPageHeader\_struct
method)
(src.software.parse.headerTelemetry.SataHostInitiatedLogPageHeader\_struct
method)
(src.software.parse.intelVUTelemetry.intelTelemetryDataAreaValidation\_union
method)
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader\_V1\_4\_struct
method)
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader\_V1\_struct
method)
(src.software.parse.intelVUTelemetry.intelTelemetryObjectHeader\_V2\_struct
method)
(src.software.parse.intelVUTelemetry.intelTelemetryTOC\_V1\_struct method)
(src.software.parse.intelVUTelemetry.intelTelemetryTOC\_V2\_struct method)
(src.software.parse.intelVUTelemetry.telemetryTObjectCatalogEntry\_struct
method)
(src.software.parse.intelVUTelemetry.telemetryTOCOBJECTEntryV2\_struct
method)
(src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1\_0\_Struct
method)
validateDataCollectConfig\(\).(src.software.guiOneShot.GUIOneShot method)
validateDownload\(\).(src.software.axon.axonInterface.Axon\_Interface method)
validateInput\(\).(src.software.userConfigurationProfile.userConfigurationProfile
method)
validation
(src.software.parse.intelVUTelemetry.intelTelemetryDataAreaValidation\_union
attribute)
validDrvIndexList\(\).(src.software.parse.internal.drive\_utility.driveList static method)
validNumericalFloat\(\).(in module src.software.JIRA.jiraEmbedding)
valueSet\(\).(src.software.guiCommon.collectGUI method)
varConstructor\(\).(src.software.datacontrol.dataControlGenMain.cUID method)
(src.software.datacontrol.dataControlGenMain.hUID method)
(src.software.datacontrol.dataControlGenMain.UID method).

```

```

varType (class in
vartypeData (src.software
(src.software
verifyAccess\(\).(src.software
version (src.software
Version (src.software
(src.software
attribute).
(src.software
attribute).
(src.software
attribute).
(src.software
attribute).
(src.software
attribute).
version1 (src.software
attribute)
Version1
(src.software.pars
attribute)
version1\_4 (src.software
attribute)
VERSION1\_TOC
(src.software.pars
version2 (src.software
attribute)
Version2
(src.software.pars
attribute)
VERSION2\_TOC
(src.software.pars
Version4
(src.software.pars
attribute)
VersionMajor
(src.software.pars
attribute)
VersionMinor
(src.software.pars
attribute)
VersionTracking
(src.software
visualizeTS (class
visualizeTS\_visua
visualizeTSAPI\(\).
vizDict (src.software
vts (src.software

```

W

wait_variable() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow
method)
 (src.software.MEP.mediaErrorGUI.StartPage method)
 (src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)
 (src.software.TSV.genericObjectGUI.StartPage method)

wait_visibility() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow
method)
 (src.software.MEP.mediaErrorGUI.StartPage method)
 (src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)
 (src.software.TSV.genericObjectGUI.StartPage method)

wait_window() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow
method)
 (src.software.MEP.mediaErrorGUI.StartPage method)
 (src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)
 (src.software.TSV.genericObjectGUI.StartPage method)

waitvar() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)
 (src.software.MEP.mediaErrorGUI.StartPage method)
 (src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)
 (src.software.TSV.genericObjectGUI.StartPage method)

WALL_CLOCK_EPOCH_TOKEN
(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog_EventHeader_union
attribute)

Warning() (src.software.parse.nlogParser.test_util.output_log.OutputLog static method)
 (src.software.parse.output_log.OutputLog static method)

warningDetected (src.software.parse.intelVUTelemetryReason.intelTelemetryReasonCode
attribute)

warningDetected()
 (src.software.parse.intelVUTelemetryReason.intelTelemetryReasonV1_0_Struct method)

warningIsError (src.software.parse.nlogParser.test_util.output_log.OutputLog attribute)
 (src.software.parse.output_log.OutputLog attribute)

warningLevel
 (src.software.parse.nlogParser.nlog_triage.hostTimeMarkerTriage.HostTimeMarkerTriage
attribute)
 (src.software.parse.nlogParser.nlog_triage.nlogTriageBase.nlogTriageBase attribute)
 (src.software.parse.nlogParser.nlog_triage.padrTriage.padrTriage attribute)
 (src.software.parse.nlogParser.nlog_triage.pssDebugTraceTriage.pssDebugTraceTriage
 attribute)

webAPI() (src.software.guiDeveloper.GUIDeveloper method)

whichScrumTeam (src.software.JIRA.class.JiraIssuesList.jira_issues_lists attribute)

WHITESPACE_TOKEN (src.software.parse.parserSrcUtil.fileTokenizer attribute)
 (src.software.parse.parserSrcUtil.tokenList attribute)
 (src.software.parse.structdefParser.structdefTokenizer attribute)

widthPredictorAPI() (src.software.autoAI.nlogPrediction.NlogPredictor method)

Window() (src.software.guiOneShot.GUIOneShot method)

WindowGenerator (class in src.software.autoAI.mediaPredictionRNN)

winfo_atom() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)
 (src.software.MEP.mediaErrorGUI.StartPage method)
 (src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)
 (src.software.TSV.genericObjectGUI.StartPage method)

winfo_atomname() (src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow
method)
 (src.software.MEP.mediaErrorGUI.StartPage method)
 (src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)
 (src.software.TSV.genericObjectGUI.StartPage method)

([src.software.TSV.genericObjectGUI.StartPage method](#))
winfo_cells().([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#)).
winfo_children().([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#)).
winfo_class().([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#)).
winfo_colormapfull().([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#)).
winfo_containing().([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#)).
winfo_depth().([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#)).
winfo_exists().([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#)).
winfo_fpixels().([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#)).
winfo_geometry().([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#)).
winfo_height().([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#)).
winfo_id().([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#)).
winfo_interps().([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#)).
winfo_vroot()
 ([src.soft](#))
 ([src.soft](#))
 ([src.soft](#))
winfo_vroot0()
 ([src.soft](#))
 ([src.soft](#))
 ([src.soft](#))
winfo_vrootx()
 ([src.soft](#))
 ([src.soft](#))
 ([src.soft](#))
winfo_vrooty()
 ([src.soft](#))
 ([src.soft](#))
 ([src.soft](#))
winfo_width()
 ([src.soft](#))
 ([src.soft](#))
 ([src.soft](#))
winfo_x().([sr](#))
 ([src.soft](#))
 ([src.soft](#))
 ([src.soft](#))
winfo_y().([sr](#))
 ([src.soft](#))
 ([src.soft](#))
withdraw().([s](#))
 ([src.soft](#))
wm_aspect().
 ([src.soft](#))
wm_attribute()
 ([src.soft](#))
wm_client().
 ([src.soft](#))
wm_colorma
 ([method](#))
 ([src.soft](#))
wm_commar
 ([src.soft](#))
wm_deiconif
 ([src.soft](#))
wm_focusmc
 ([src.soft](#))
wm_forget()
 ([src.soft](#))
wm_frame()
 ([src.soft](#))
wm_geometr
 ([src.soft](#))
wm_grid().([s](#))
 ([src.soft](#))

[\(src.software.TSV.genericObjectGUI.StartPage method\)](#)
[winfo_ismapped\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[winfo_manager\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[winfo_name\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[winfo_parent\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[winfo_pathname\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[winfo_pixels\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[winfo_pointerx\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[winfo_pointery\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[winfo_pointerxy\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[winfo_pointery\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[winfo_reqheight\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[winfo_reqwidth\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))
 ([src.software.MEP.mediaErrorGUI.StartPage method](#))
 ([src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method](#))
 ([src.software.TSV.genericObjectGUI.StartPage method](#))
[winfo_rgb\(\)](#)([src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method](#))

[wm_group\(\)](#)
 ([src.soft](#)
[wm_iconbitn](#)
 ([src.soft](#)
[wm_iconify\(\)](#)
 ([src.soft](#)
[wm_iconmas](#)
 ([src.soft](#)
[wm_iconnam](#)
 ([src.soft](#)
[wm_iconpho](#)
 ([src.soft](#)
[wm_iconposi](#)
 ([src.soft](#)
[wm_iconwin](#)
 ([src.soft](#)
[wm_manage](#)
 ([src.soft](#)
[wm_maxsize](#)
 ([src.soft](#)
[wm_minsize](#)
 ([src.soft](#)
[wm_override](#)
 ([src.soft](#)
[wm_position](#)
 ([src.soft](#)
[wm_protocol](#)
 ([src.soft](#)
[wm_resizable](#)
 ([src.soft](#)
[wm_sizefrom](#)
 ([src.soft](#)
[wm_state\(\)](#)
 ([src.soft](#)
[wm_title\(\)](#)
 ([src.soft](#)
[wm_transient](#)
 ([src.soft](#)
[wm_withdraw](#)
 ([src.soft](#)
[word\(\)](#)([src.so](#)
 ([src.soft](#)
[writeARMA](#)
[writeBlock\(\)](#)
[writeFile\(\)](#)([\\$](#)
[writeINIFile](#)
[writeMetaPr](#)
[WriteNlogSt](#)
 ([src.software](#)
[method](#))
[writeOneSho](#)
[writeRNtoI](#)
[WriteTriageS](#)
 ([src.software](#)
[method](#))

<p>(src.software.MEP.mediaErrorGUI.StartPage method)</p> <p>(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)</p> <p>(src.software.TSV.genericObjectGUI.StartPage method)</p> <p>winfo_rootx(),(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)</p> <p>(src.software.MEP.mediaErrorGUI.StartPage method)</p> <p>(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)</p> <p>(src.software.TSV.genericObjectGUI.StartPage method)</p> <p>winfo_roots(),(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)</p> <p>(src.software.MEP.mediaErrorGUI.StartPage method)</p> <p>(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)</p> <p>(src.software.TSV.genericObjectGUI.StartPage method)</p> <p>winfo_screen(),(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)</p> <p>(src.software.MEP.mediaErrorGUI.StartPage method)</p> <p>(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)</p> <p>(src.software.TSV.genericObjectGUI.StartPage method)</p> <p>winfo_screencells(),(src.software.MEP.mediaErrorGUI.GenericObjectAppMainWindow method)</p> <p>(src.software.MEP.mediaErrorGUI.StartPage method)</p> <p>(src.software.TSV.genericObjectGUI.GenericObjectAppMainWindow method)</p> <p>(src.software.TSV.genericObjectGUI.StartPage method)</p>	<p>writeTSVVisu</p> <p>method)</p> <p>(src.soft</p>
--	---

X

<p>XlateAndOutputMultipleNlog()</p> <p>(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.TelemetryV2NlogEventParserL2 method)</p> <p>XlateAndOutputNlog()</p> <p>(src.software.parse.nlogParser.telemetry_parsers.telemetry_nlog.TelemetryV2NlogEventParserL2 method)</p>	<p>xlateEv</p> <p>(src.sof</p> <p>xlateTo</p> <p>(src.sof</p> <p>method</p> <p>xmlFile</p>
---	--

Z

<p>zipFile((src.software.guiOneShot.GUIOneShot attribute))</p>	<p>zipName ((src.software.guiOneShot.GUIOneShot attribute))</p>
--	---

Linux Ubuntu Mate x64

Anaconda Installation

1. Download file

```
wget https://repo.anaconda.com/archive/Anaconda3-  
2020.07-Linux-x86_64.sh  
chmod 755 Anaconda3-2020.07-Linux-x86_64.sh  
. ./Anaconda3-2020.07-Linux-x86_64.sh -b -p  
$HOME/anaconda3
```

2. Open Anaconda

3. Change to repository directory

4. Recreate and update the base environment with the best known method.

- Please note `environment_ubuntu-x86_64.yml` should be `environment_{operating system}.yml`

```
conda update --force conda -y  
conda update anaconda -y  
conda update --all  
conda update anaconda-navigator  
conda update python  
conda env create --file environment_ubuntu-x86_64.yml
```

5. To update use

```
conda env update -f environment_ubuntu-x86_64.yml
```

Jet Brains PyCharm Debugger

- <https://www.jetbrains.com/pycharm/>

```
sudo snap install pycharm-community --classic
```

Windows 10 x64

Installation

- Open command prompt

```
set url=https://repo.anaconda.com/archive/Anaconda3-  
2020.07-Windows-x86_64.exe  
set file=Anaconda3-2020.07-Windows-x86_64.exe  
certutil -urlcache -split -f %url% %file%  
start /wait "" Anaconda3-2020.07-Windows-x86_64.exe  
/InstallationType=JustMe /RegisterPython=0 /S  
/D=%UserProfile%\anaconda3  
start /wait ""  
%UserProfile%\anaconda3\Scripts\activate.bat
```

- Anaconda should be open or open Anaconda manually
- Change to repository directory
- Recreate and update the base environment with the best known method.
 - Please note `environment_windows-x86_64.yml` should be `environment_{operating system}.yml`

```
conda update --force conda -y  
conda update anaconda -y  
conda update --all  
conda update anaconda-navigator  
conda update python  
conda env create --file environment_win-x86_64.yml
```

- To update use:

```
conda env update -f environment_win-x86_64.yml
```

Pycharm Debugger

Client Config via Command Line

- Create file: silent.config using create config file via command line

```
echo. > "silent.config"
echo mode=user >> "silent.config"
echo launcher32=0 >> "silent.config"
echo launcher64=1 >> "silent.config"
echo updatePATH=0 >> "silent.config"
echo jre32=1 >> "silent.config"
echo updateContextMenu=1 >> "silent.config"
echo python2=0 >> "silent.config"
echo python3=0 >> "silent.config"
echo regenerationSharedArchive=1 >> "silent.config"
echo .py=0 >> "silent.config"
```

- Optionally, edit config file manually:

```
; Installation mode. It can be user or admin.
; NOTE: for admin mode please use "Run as
Administrator" for command prompt to avoid UAC dialog
or user 'admin'.
mode=user

; Desktop shortcut for launchers
launcher32=0
launcher64=1

; Add launchers path to PATH env variable
updatePATH=0

; Download and install jre32. This may take a few
minutes.
jre32=1

; Add "Open Folder as Project" to context menu
updateContextMenu=1

; Download and install python. This may take a few
minutes.
python2=0
```

```
python3=0

; Regenerating the Shared Archive
; https://docs.oracle.com/en/java/javase/11/vm/class-
data-sharing.html
regenerationSharedArchive=1

; List of associations. To create an association change
value to 1.
.py=0
```

Commandline Install

- Installing in command line

```
set url=https://download.jetbrains.com/python/pycharm-
community-2021.2.2.exe
set file=pycharm-community-2021.2.2.exe
certutil -urlcache -split -f %url% %file%
start /wait "" pycharm-community-2021.2.2.exe /S
/CONFIG=.silent.config
/LOG=C:\JetBrains\PyCharmEdu\install.log
/D=C:\JetBrains\Edu\PyCharm_2020
```

RAAD Executable Installer for Windows 10 x86_64

If it is your first time utilizing RAAD, please follow the instructions below to download and execute the required files to start developing or utilizing RAAD. There are two sections in this article:

- one for developers.
- one for users of RAAD.

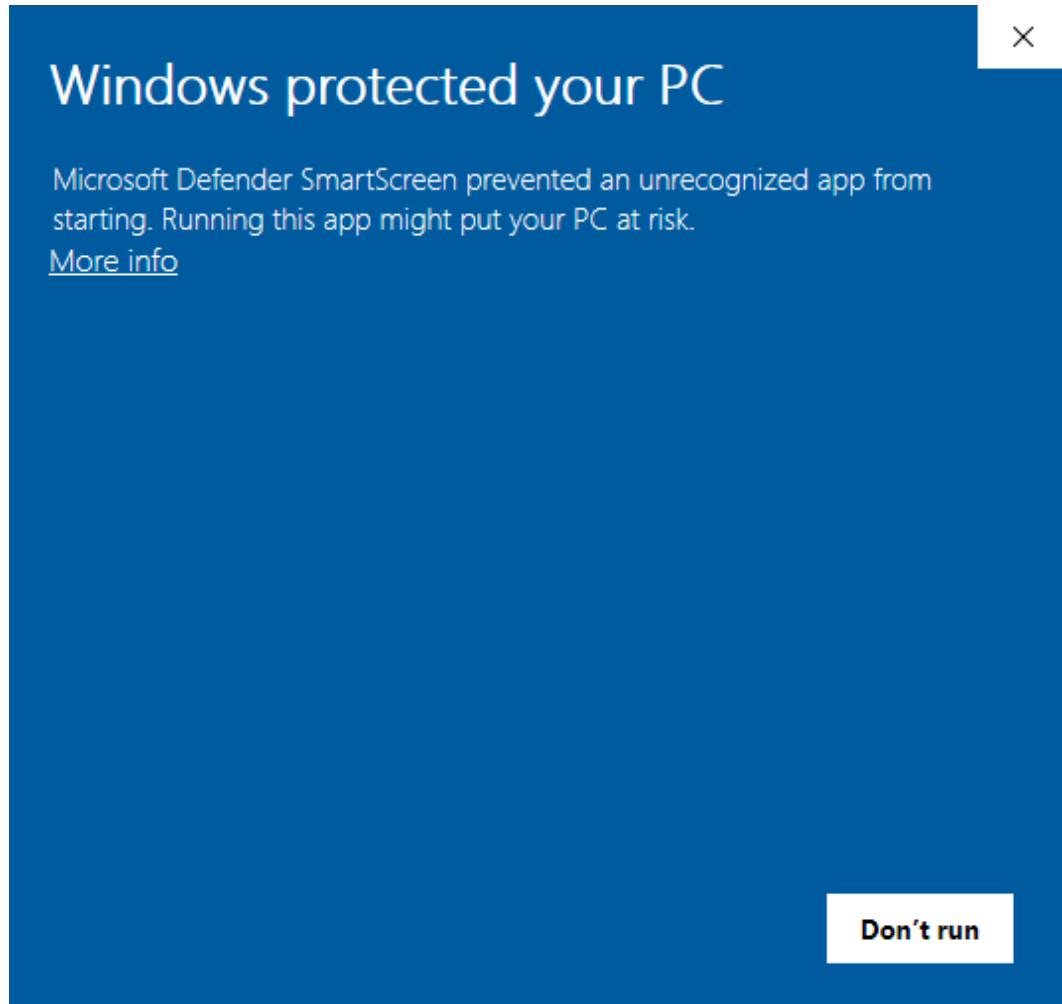
The section for developers is directly below. The section for users can be found after the section for developers. Please note the tutorial is for Windows, the tutorial for Ubuntu/Linux will be uploaded at a later iteration of the project.

Information for Developers of RAAD:

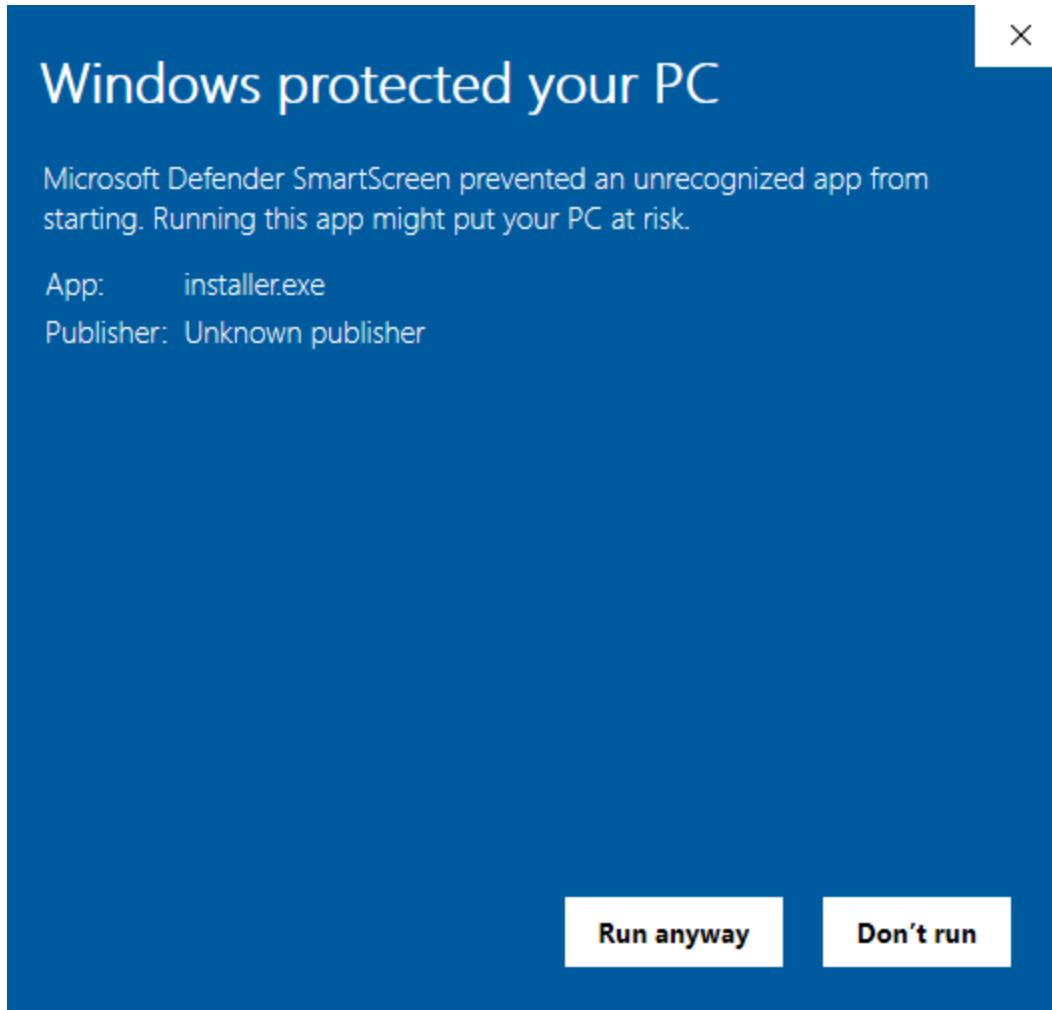
- As a developer you need to download the installation wizard found in the RAAD under "RAAD Executables": The wizard is named installer.exe. Once you have downloaded the file from teams, please follow the instructions below. @todo jdtarang

Instructions for the installation wizard:

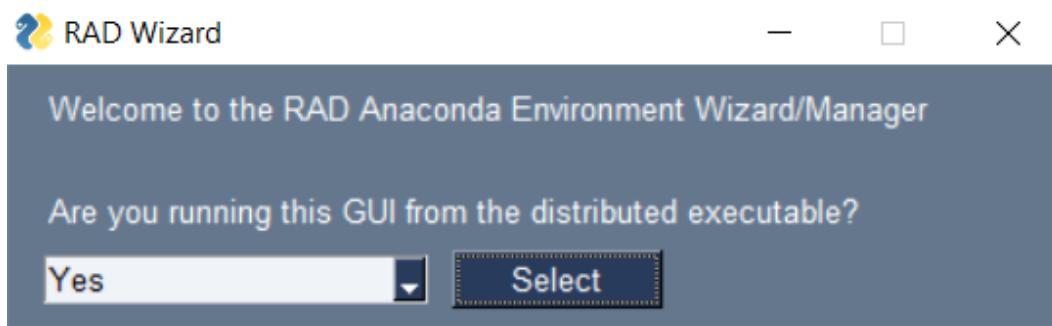
After the download is done, run the executable. You may encounter a blue window telling you "Windows protected your PC".



Click on "More info" and then on the new button "Run Anyway".

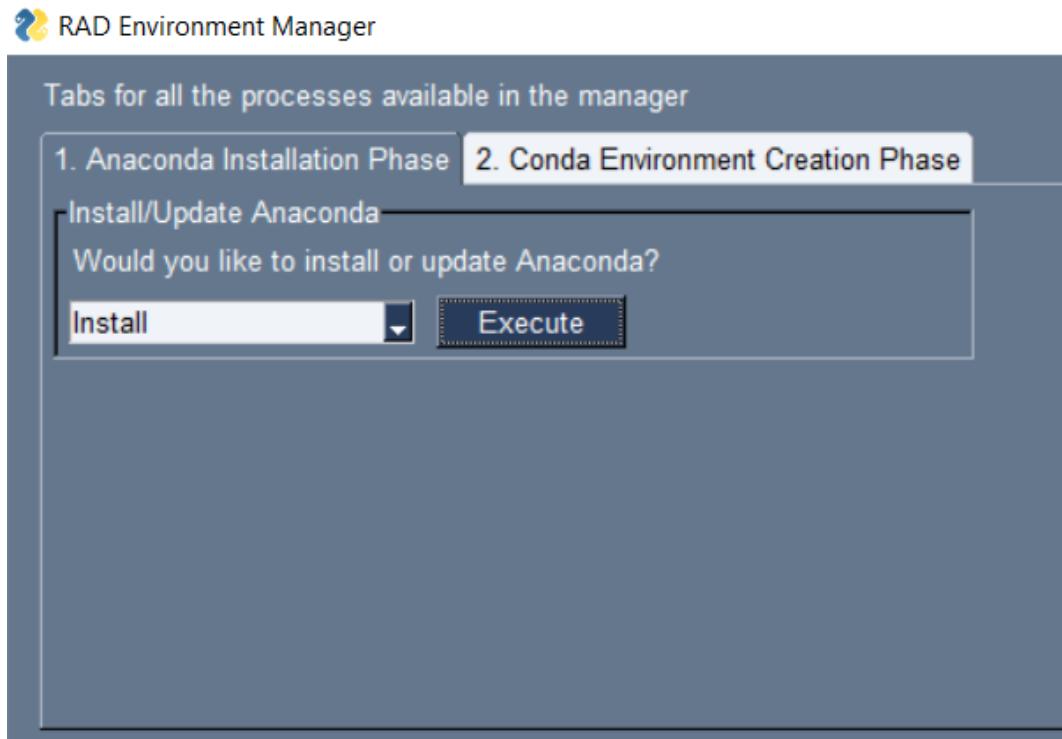


Once the executable is running, you should see a background terminal window with all the warning and info statements for the wizard executable. If the executable runs correctly, a window like the one shown below should pop-up on top of the terminal window. Select "Yes" on the first drop down menu (as the wizard is being run directly from the executable), and continue with the other steps.

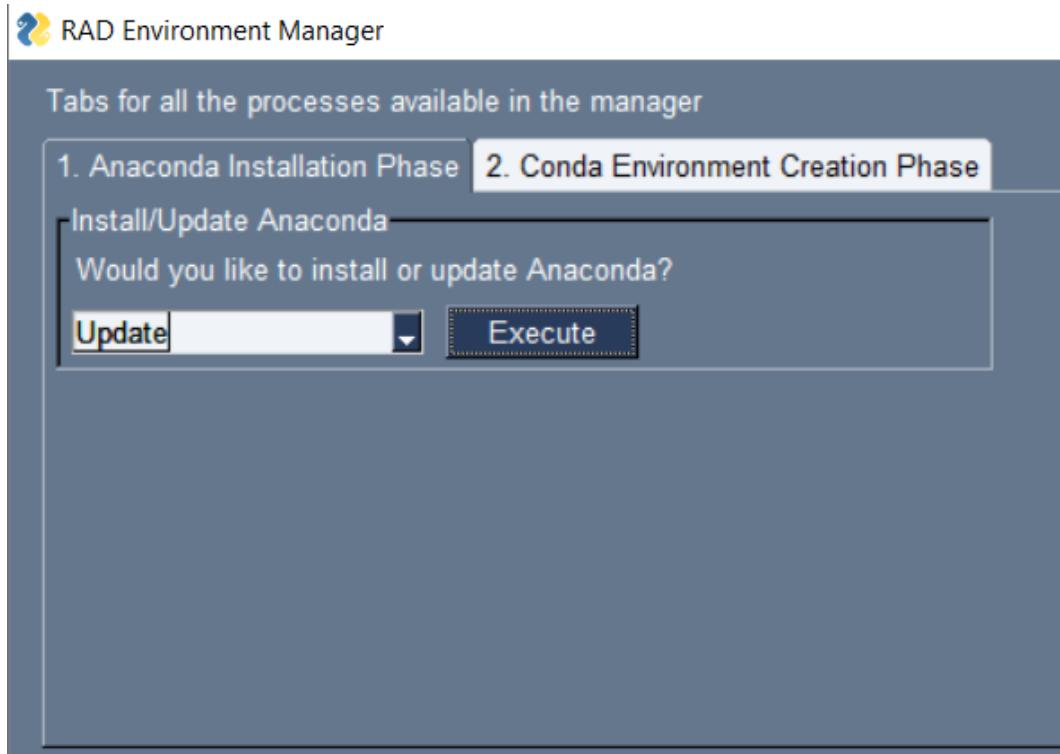


Note: Make sure you click on "Select" so that the GUI registers the option chosen in the drop down menu.

After you have accessed the second screen, install Anaconda by choosing the "install" option from the drop down menu and clicking the "Execute" button shown below. The Anaconda executable will be downloaded and the installer will be launched. Accept the terms and conditions and don't change any of the options while installing anaconda (click on next until you reach the installation). Wait for the installation to finish and then return to RAAD installation Wizard



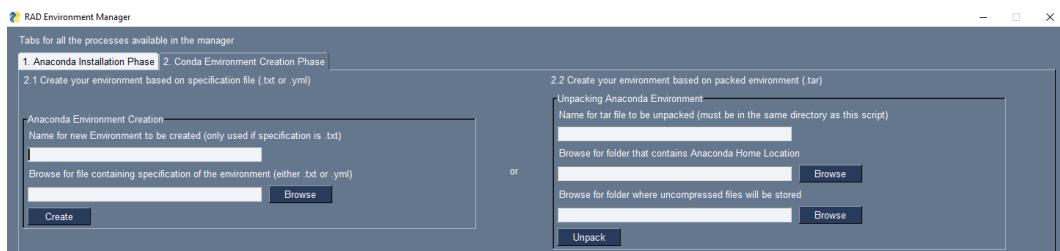
Note: If for some reason you already had Anaconda installed in your device, you can update it to the latest version by choosing the "update" option in the drop down menu and clicking the "Execute" button shown below.



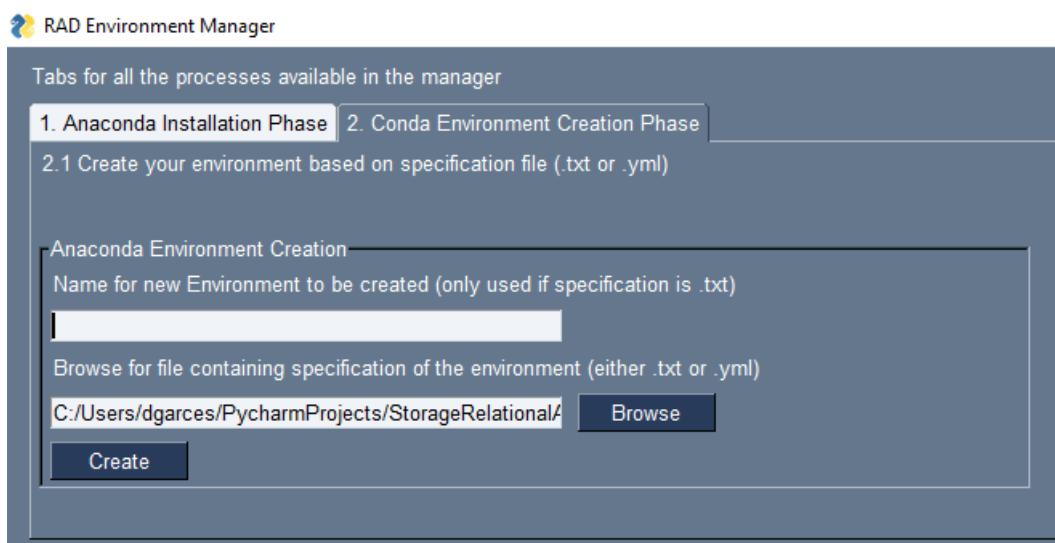
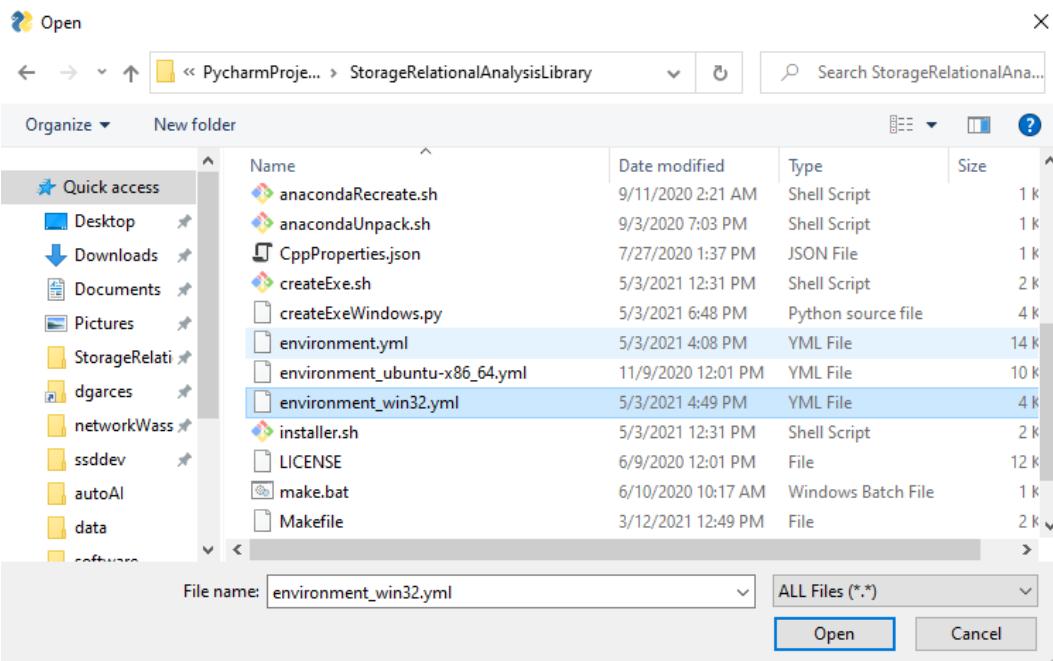
Note: If the wizard starts crashing before Anaconda gets updated/installed, please check your connection to the internet and try again. If the problem persists, please manually download anaconda by going to
<https://www.anaconda.com/products/individual>

Download the environment configuration file found here:
`environment_win-x86_64.yml`

After the file has been download into your local machine, click on the tab titled "2. Conda Environment Creation Phase" to change to the next tab and start the environment creation process. The tab looks like the one displayed below



Once you are in the "Conda Environment Creation Phase" tab, use the "Anaconda Environment Creation" tool to create the environment using the .yml that you downloaded in step 3. To create the environment, you need to click on the "Browse" button, which will display a navigator window as shown below. Use this navigation window to locate your copy of "environment_win32.yml" and click the open button to insert the path in the input line.

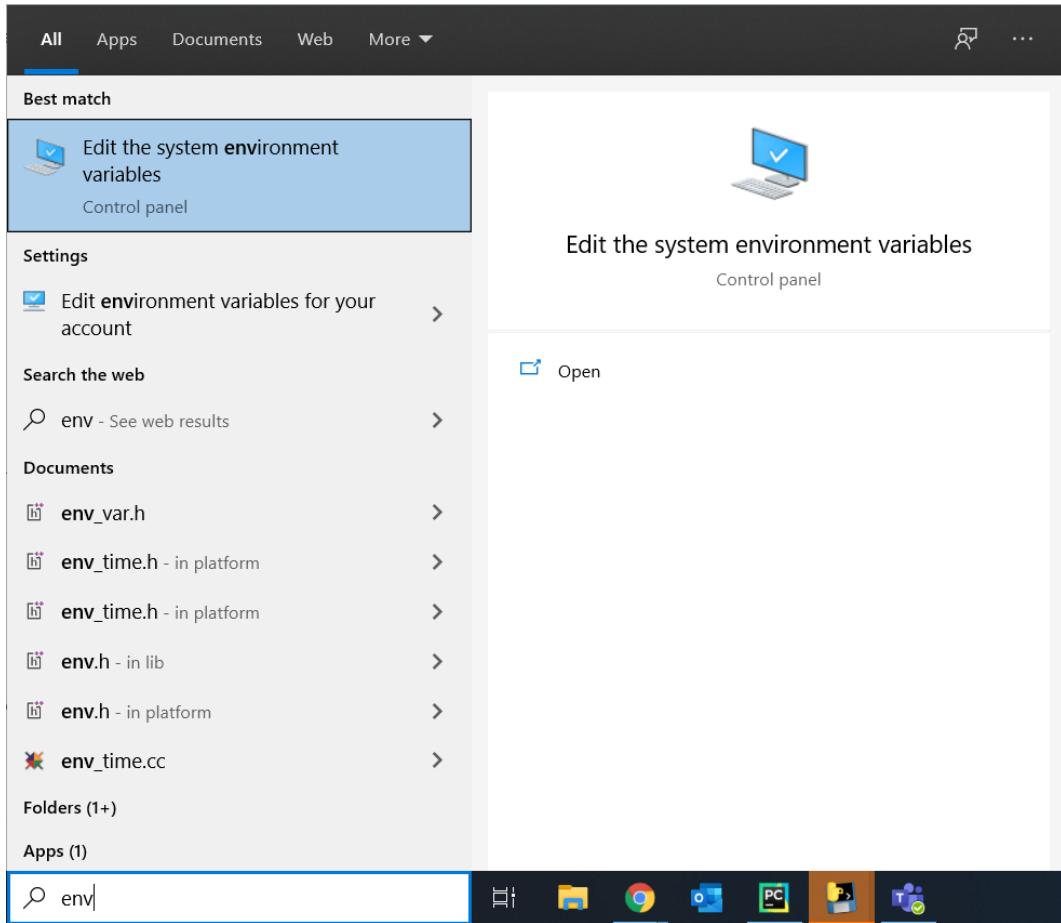


Once the path is in the input line, click on the create button and wait for the process to finish. If the environment was created successfully, a success message like the one displayed below will be shown in the background terminal window.

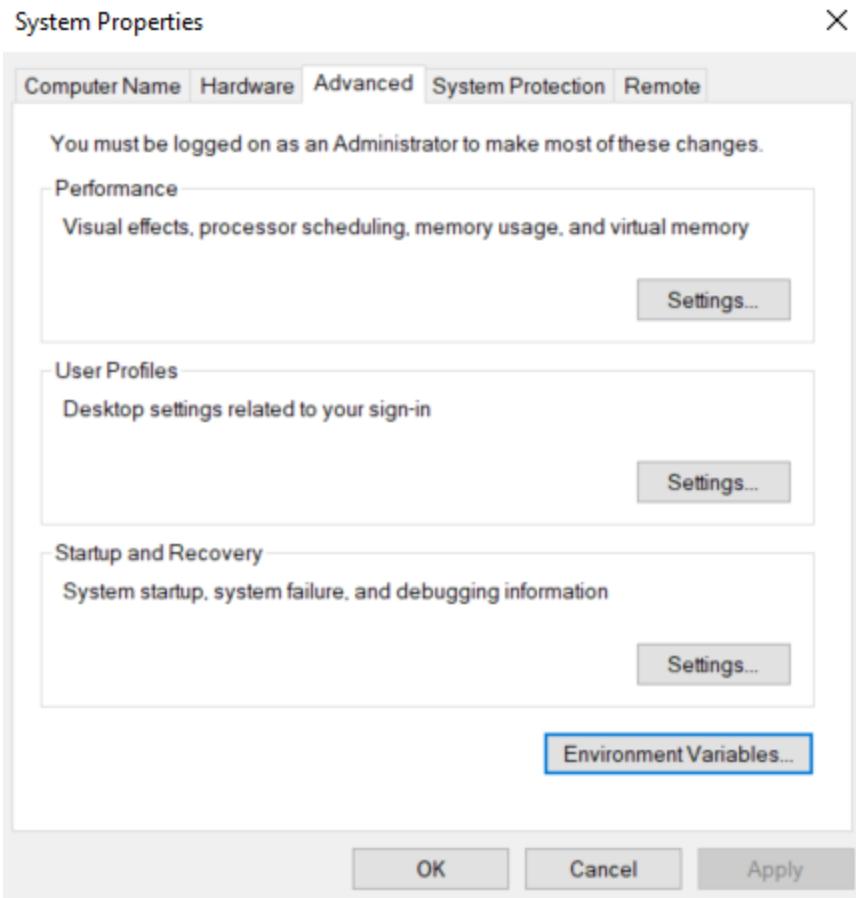
```
Successfully installed abs1-py-0.11.0 alabaster-0.7.12 altgraph-0.17 asn1crypto-1.4.0 astunparse-1.6.3 babe
1-2.9.0 cachetools-4.1.1 cffi-1.14.5 chardet-3.0.4 colorama-0.4.4 cryptography-3.4.7 docutils-0.16 flatbuff
ers-1.12 future-0.18.2 gast-0.3.3 gnupg-2.3.1 google-auth-1.23.0 google-auth-oauthlib-0.4.2 google-pasta-0.
2.0 grpcio-1.32.0 h5py-2.10.0 idna-2.10 imagesize-1.2.0 importlib-metadata-2.0.0 jinja2-2.11.3 keras-prepro
cessing-1.1.2 markdown-3.3.3 markupsafe-1.1.1 naked-0.1.31 numpy-1.19.5 oauthlib-3.1.0 opt-einsum-3.3.0 ord
ered-set-4.0.2 packaging-20.9 pfile-2019.4.18 protobuf-3.13.0 pyasn1-0.4.8 pyasn1-modules-0.2.8 pycparser-
2.20 pycryptodome-3.10.1 pygments-2.8.0 pyinstaller-hooks-contrib-2021.1 pylatex-1.4.1 pwini
n32-3.0.0 pywin32-ctypes-0.2.0 pyyaml-5.3.1 requests-2.25.0 requests-oauthlib-1.3.0 rsa-4.6 shellescape-3.8.1
snowballstemmer-2.1.0 sphinx-3.5.1 sphinxcontrib-applehelp-1.0.2 sphinxcontrib-devhelp-1.0.2 sphinxcontrib-
htmlhelp-1.0.3 sphinxcontrib-jsmath-1.0.1 sphinxcontrib-qthelp-1.0.3 sphinxcontrib-serializinghtml-1.1.4 t
ensorboard-2.4.0 tensorflow-plugin-wit-1.7.0 tensorflow-2.4.1 tensorflow-estimator-2.4.0 termcolor-1.1.0 t
inyaes-1.0.1 typing-extensions-3.7.4.3 urllib3-1.26.1 werkzeug-1.0.1 wrapt-1.12.1 zipp-3.4.0
done
#
# To activate this environment, use
#
#     $ conda activate RAD2.0
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

Note: The main wizard GUI will be frozen while this process executes. Look at the background terminal window to check for progress, but do not close the wizard window as this will terminate the process and might affect future installation attempts.

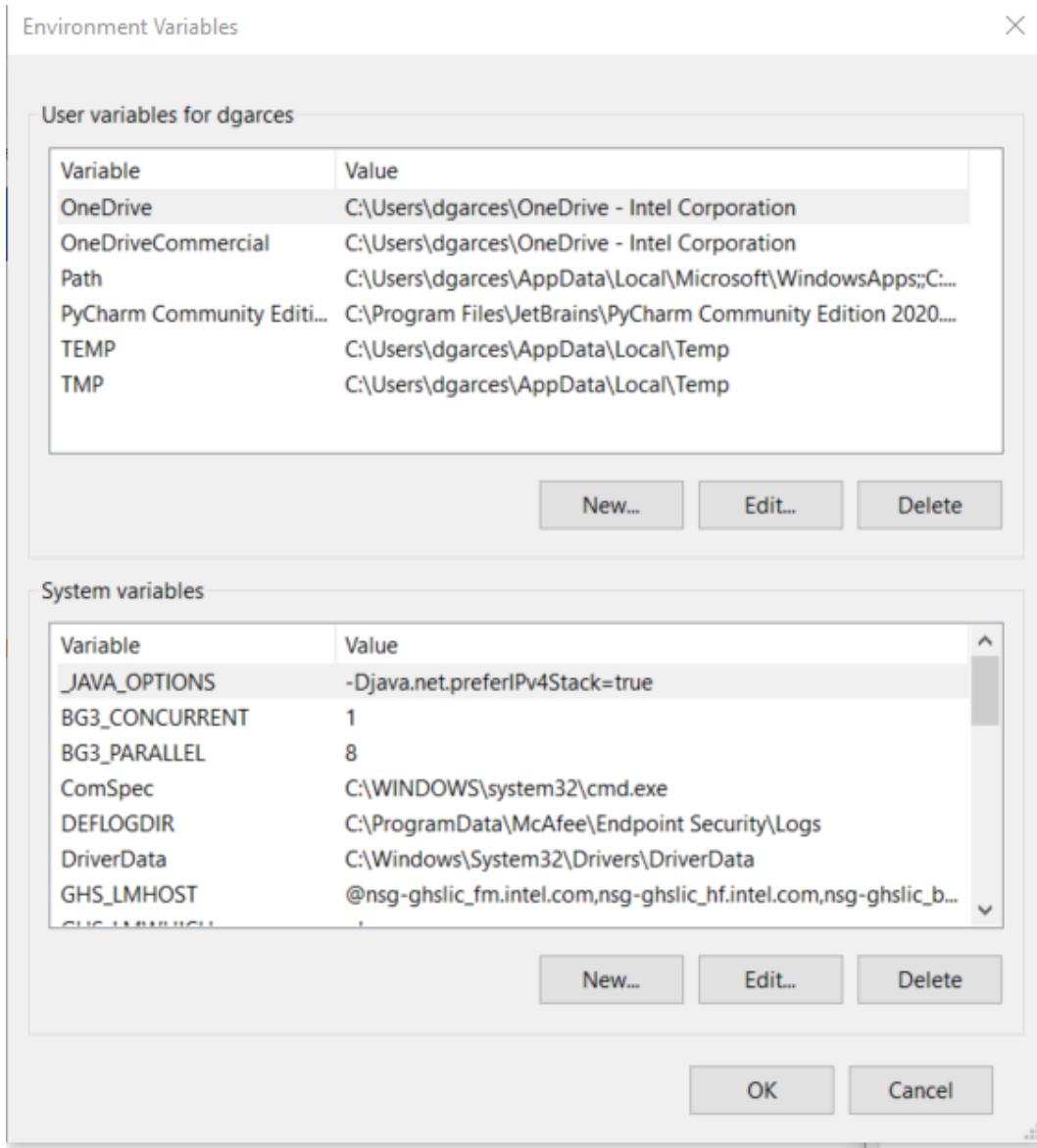
If you find a proxy error, please follow the following instructions, which are based on the initial solution proposed by Joe Tarango here: - Look for env in your search bar near the lower left of your screen. Go to "Edit the System Environment Variables" as shown below.



- Once this configuration opens, click "Yes" on the pop-up window.



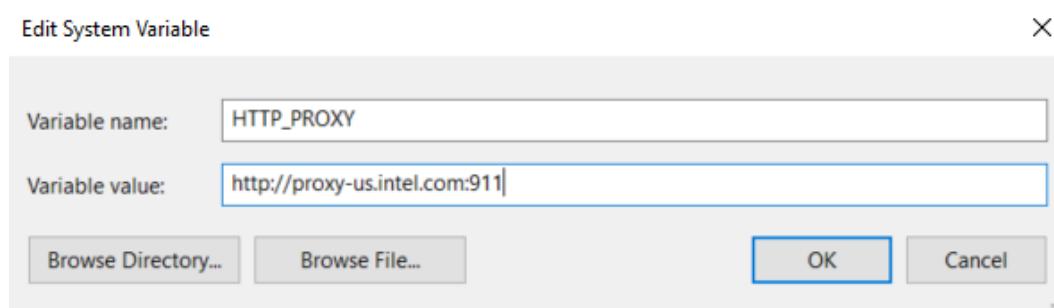
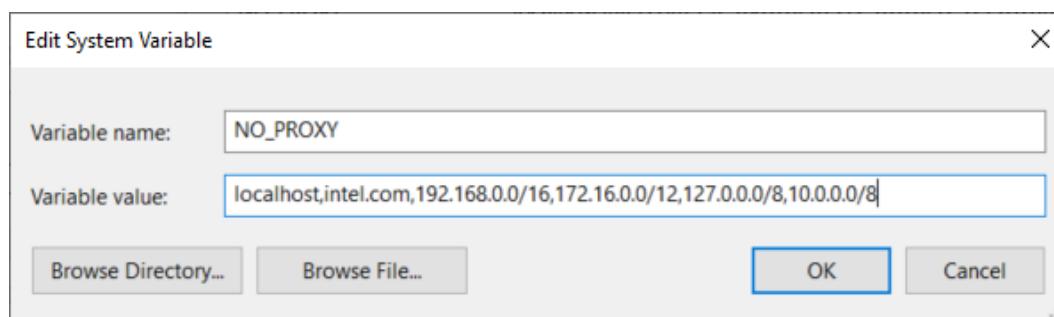
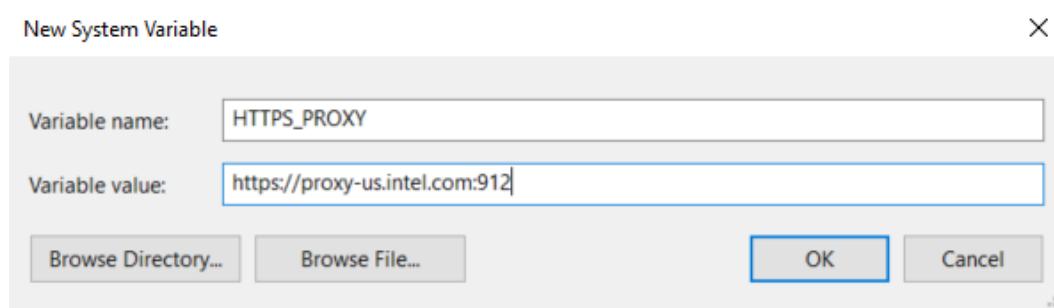
- Click on "Environment Variables" as shown below.



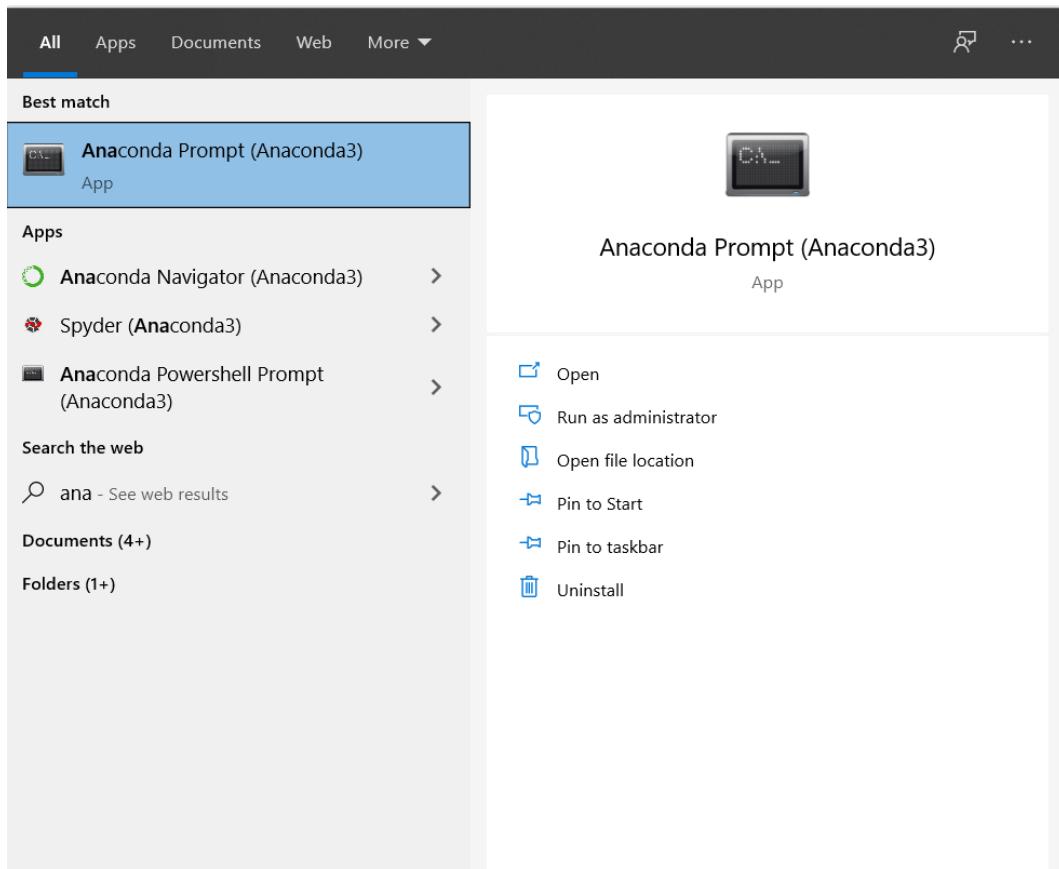
- Click on the "New..." button for System variables and then add the following 3 variables as shown in the pictures below. Please consider that the examples below are for configuring the proxy with the US network. Change your configuration accordingly to reflect the location of the proxy that you want to utilize. Please refer to the code below for the easy cut and paste variable values.

```
HTTPS_PROXY  
https://proxy-chain.com:2  
NO_PROXY  
localhost,192.168.0.0/16,172.16.0.0/12,127.0.0.0/8,10.0.0.0/8
```

HTTP_PROXY
http://proxy-chain.com:1



- Open an Anaconda3 Prompt (it should have been installed during the Anaconda installation), and type: where .condarc.



Note: If there is no .condarc, please create this file inside C:Users<your-user>, where <your-user> corresponds to the user name that you are utilizing on your local machine.

A screenshot of an Anaconda Prompt window. The title bar says "Anaconda Prompt (Anaconda3)". The command line shows the following text:

```
(base) C:\Users\dgarces>where .condarc
C:\Users\dgarces\.condarc

(base) C:\Users\dgarces>
```

The text is displayed in white on a black background.

- Navigate to the file location and open the file with your editor of choice (emacs, vim, notepad++, etc). Modify your file to resemble the file shown below.

```
channels:
  - conda-forge
  - defaults
  - intel
  - pytorch
  - anaconda
  - bioconda
  - mkl

ssl_verify: true
allow_other_channels: true

# Proxy settings: http://[username]:[password]@[server]:[port]
proxy_servers:
http: http://proxy-chain.com:1
https: https://proxy-chain.com:2

# Implies always using the --yes option whenever asked to
proceed
always_yes: true

# Auto updating of dependencies
update_dependencies: true

# Environment variables to add configuration to control the
number of threads. Choose for you machine.
default_threads: 4

# Update conda automatically
auto_update_conda: true

# Enable certain features to be tracked by default.
track_features:
- mkl

# pip_interop_enabled (bool)
#   Allow the conda solver to interact with non-conda-installed
python packages.
pip_interop_enabled: true

# Show channel URLs when displaying what is going to be
downloaded.
show_channel_urls: true
```

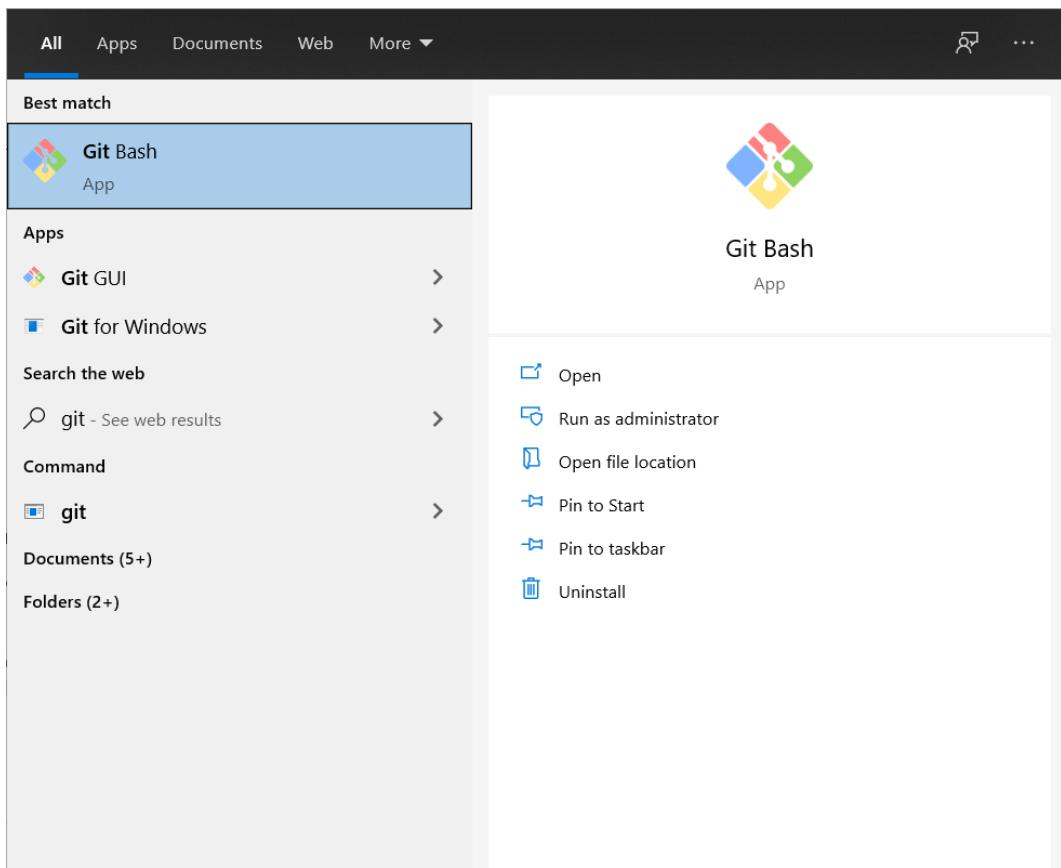
```
# Opt in, or opt out, of automatic error reporting to core
# maintainers.
# Error reports are anonymous, with only the error stack
# trace and information given by `conda info` being sent.
report_errors: false
```

- If you don't have it installed, please install the C++ Redistributable included in the Visual Studio 2019, which can be found here: <https://visualstudio.microsoft.com/downloads/>. After the download and the installation. Close all the installers and restart your computer. After your computer reboots, open the RAAD installation wizard again and try the environment creation again.
- If the issues persist, uninstall anaconda by following the instructions in <https://docs.anaconda.com/anaconda/install/uninstall/>, kill the RAAD installation Wizard and start again from step 1.
- If you face additional issues with anaconda not described in this tutorial, please use the following list to manually install the packages in the conda prompt. Navigate to Anaconda Prompt and execute each of the following commands independently. Wait for the command to complete its execution before running the next command

```
conda create --name raad python=3.8
conda activate raad
conda uninstall pip
conda install pip=20.2.4=py38_0
conda install wgetter
conda install pandas=1.0.5
conda install -c conda-forge scikit-learn
conda install statsmodels
conda install -c anaconda psutil
conda install -c conda-forge gputil
conda install -c anaconda cryptography
conda install -c anaconda pycrypto
conda install -c anaconda urllib3
conda install -c conda-forge atlassian-python-api
conda install -c conda-forge jira
conda install -c conda-forge pyinstaller
pip install matplotlib==3.3.1
pip install tensorflow==2.3.0
pip install gnupg
```

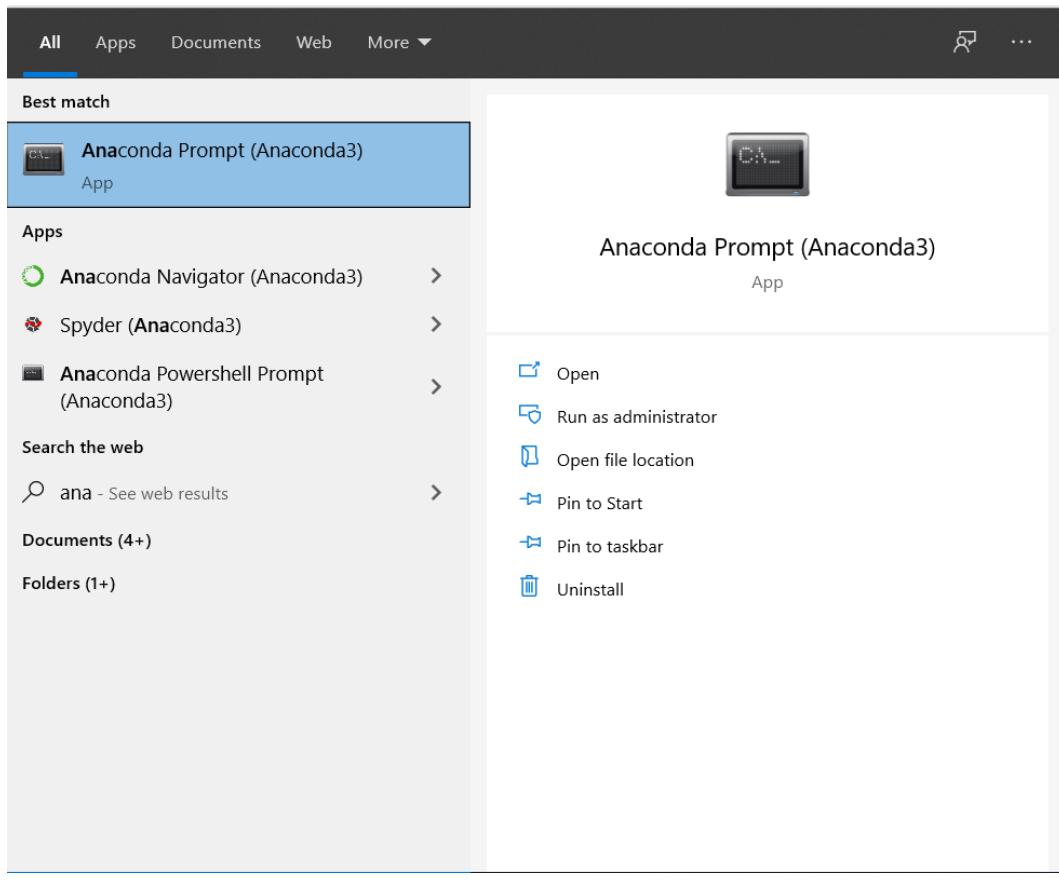
```
pip install PyLaTeX  
pip install tinyaes  
pip install PySimpleGUI  
pip install tornado  
pip install pycairo  
pip install unidecode  
pip install sentence-transformers
```

- After the creation of the environment finishes, you need to clone the RAAD repository by using git. If you do not have git, please go here to download it. Open a git terminal (as shown below), and execute the following command:

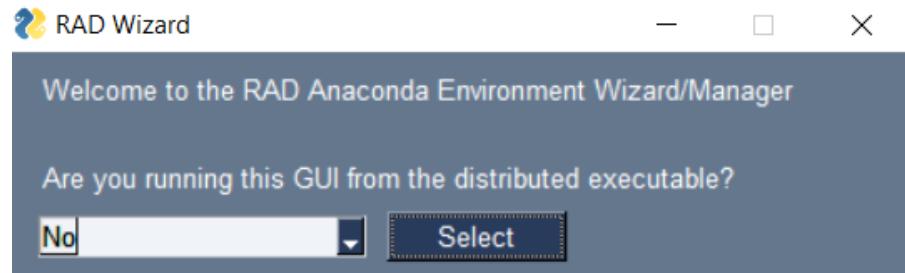


```
git clone https://github.com/Intel/RAAD.git
```

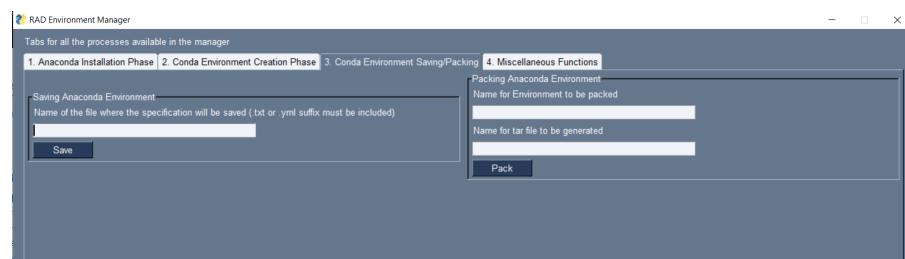
- After the repo has been cloned, close the installation wizard and the git bash terminal. Open an Anaconda3 Prompt (it should have been installed during the Anaconda installation) and activate the environment by executing conda activate RAAD2.0



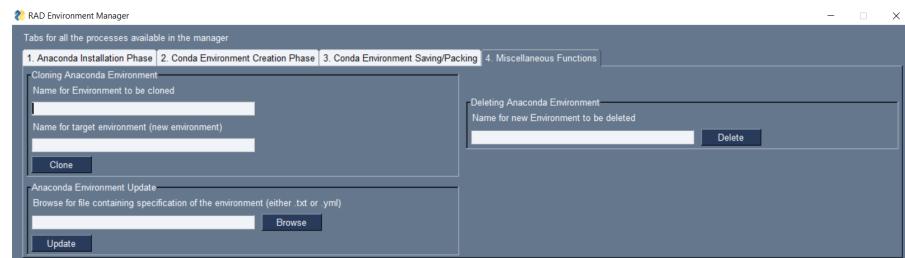
- Use the anaconda prompt to navigate to the location of the recently cloned "RAAD" repo and start developing using emac, vim, or your IDE of choice (I personally recommend PyCharm, which has a ton of useful features for developing in Python). If you decide to use PyCharm, you might need to open the repo using their interface, so it is properly initialized as a PyCharm project and you can use the embedded terminal for git and executing your scripts.
- If you want to easily manage your environments, navigate inside the repo to src/ and then run the installer GUI by utilizing the following command once you are inside src: python installer.py



- Select "No" in the first window for the installation wizard



- utilize the other miscellaneous functionalities that we have included to facilitate environment management for RAAD.



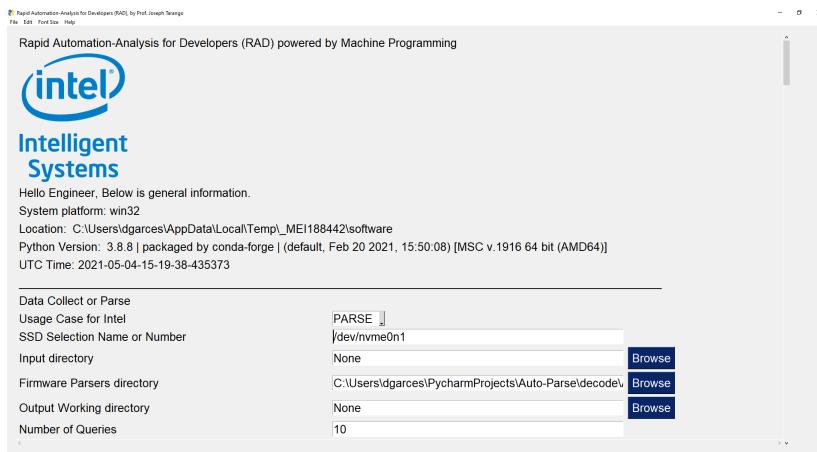
Information for Users of RAAD

As an user of RAAD, you need to access the GUI to run the functionality of our system,. There are two choices for running the GUI:

- Follow the instructions for developers all the way to steps above. Once you have a copy of the repo in your machine, run the main.py script by executing the following command from Anaconda3 prompt (with the right environment activated), or from your IDE of choice (as long as you are using the right environment for the interpreter) - please remember that you need

to navigate to src/ before running the command: python main.py --mode 1

- After around 30 seconds of automatic set-up, the system will generate a window like the one displayed below. Follow the subsection "Instructions for the GUI" for more information on how to operate the GUI.



- Create or use the GUI executable. @todo jdtarang folder inside the RAAD teams channel: The GUI is named gui.exe. Once you have the executable and its accompanying folder, make sure you save them both to the same location in your machine to allow the executable to operate correctly. The software file should contain a single file (the logo for the GUI), please make sure the logo is directly inside "software" and not nested inside other folders. To open and run the executable, just double-click on it.
 - You may encounter a blue window telling you "Windows protected your PC". Click on "More info" and then on the new button "Run Anyway"



- Follow the subsection "Instructions for the GUI" for more information on how to operate the GUI.



Note: Method 1 is preferable to guarantee that you get all the latest changes and recent code pushes that are constantly being added to the repo to improve the system functionality; however, method 2 provides a stable baseline with the basic functionality at the time of writing.