Welcome!						
This document contains information about all the classes in the Python-POD_API.	Each class has its own sheet which describes the class's origin, parent and child classes, imports, global and instance variables, and all methods with details about its parameters, returns, and exceptions.					
Classes						
Setup_8206HR						
POD_8206HR						
POD_8401HR						
POD_Basics						
POD_Packets						
POD_Commands						
COM_io						

Class						
Name	**File**	**Description**	**Parent**	**Child**	**Author**	
Setup_8206HR	CodeSetup_8206HR.py	Allows a user to set up and stream from any number of 8206HR POD devices. The streamed data is saved to a file.	N/A	N/A	Thresa Kelly	
Imports						
Name	**Origin**	**Description**	**From**			
COM_io	Local	Used to get the COM ports list	SerialCommunication			
POD_8206HR	Local	For managing active 8206HR POD devices	PodDevice_8206HR			
texttable	Enviornment	Used to generate a table to display the POD parameters	N/A			
threading	Enviornment	Used to stream from multiple POD devices and ask for user input concurrently.	N/A			
time	Enviornment	For timing the duration of methods	N/A			
numpy	Enviornment	For arrays and data handling	N/A			
path	Enviornment	Used for file handling	os			
EdfWriter	Enviornment	For writing to EDF files	pyedflib			
Variables						
Name	**Scope**	**Description**	**Value**			
_PHYSICAL_BOUND_uV	Class	Physical min/max for EDF file	4069			
_podDevices	Instance	Dictionary containing all POD_8206HR class objects	Set by user			
_podParametersDict	Instance	Dictionary containing the setup parameters for all POD_8206HR objects	Set by user			
_saveFileName	Instance	String containing the path, filename, and file extension to a file to save streaming data to. The filename will be extended with "_<DEVICE NUMBER>" for each device.	Set by user			
_options	Instance	Dictionary listing the different options for the user to complete	{ 1 : 'Start Streaming.', 2 : 'Show current settings.', 3 : 'Edit save file path.', 4 : 'Edit POD device parameters.', 5 : 'Connect a new POD device.', 6 : 'Reconnect current POD devices.', 7 : 'Generate initialization code.', 8 : 'Quit.' }			
Methods						
Name	**Type**	**Description**	**Parameter Name**	**Parameter Purpose**	**Return**	**Exception**
SetupPODparameters	Instance	Prompts user for the POD device parameters if none are provided. Used in initialization.	podParametersDict=None	Dictionary containing parameters to setup a POD device	N/A	N/A
SetupSaveFile	Instance	Asks user for a file path and name to save the POD stream data to if not provided. Used in initialization.	saveFile=None	String describing the file path and file name with extension	N/A	N/A
Run	Instance	Gives the user options until quitting.	N/A	N/A	N/A	N/A
_SetNumberOfDevices	Static	Asks the user how many POD devices they want.	N/A	N/A	Integer number for the number of devices	The number must be an integer greater than zero.
_DisconnectAllPODdevices	Instance	deletes all POD devices	N/A	N/A	N/A	N/A
_ConnectAllPODdevices	Instance	Connects all pod devices after disconnecting previous devices	N/A	N/A	N/A	N/A
_ConnectPODdevice	Instance	Creates a POD_8206HR object and write the POD parameters to the device.	deviceNum	Unique dictionary key representing the POD device	True when the device is successfully connected, false otherwise.	N/A
			deviceParams	dictionary containing parameters to setup a POD device		
_AddPODdevice	Instance	Generates new device number and asks the user for the POD device parameters.	N/A	N/A	N/A	N/A
_SetParam_allPODdevices	Instance	Asks user for the number of POD devices and requests the parameters for all devices.	N/A	N/A	N/A	N/A
_GetParam_onePODdevice	Static	Creates dictionary from POD device parameters user input	N/A	N/A	Dictionary containing the POD parameters for one device (port, sample rate, preamplifier gain, and low pass)	N/A
_ChoosePort	Static	Gets a list of available ports and asks user which port to connect to the current POD device.	forbidden=[]	List of String containing the names of ports not allowed to be connected to	String name of the chosen port	The user input must match an allowed COM port
_GetPortsList	Static	Gets a list of available port names. If there are no ports, the program will wait until another POD device is plugged in.	forbidden=[]	List of String containing the names of ports not allowed to be connected to	List of strings containing the name of all COM ports.	There must be COM ports in use
_ChooseSampleRate	Static	Asks user for the sample rate.	N/A	N/A	Integer number between 100-2000 Hz for the sample rate	Sample rate must be an integer between 100-2000
_ChoosePreampGain	Static	Asks user for the preamplifier gain of their POD device	N/A	N/A	Integer 10 or 100 for the preamplifier gain	Gain must be an integer value of 10 or 100
_ChooseLowpass	Static	Builds dictionary of all lowpass filters	N/A	N/A	Dictionary containing lowpass filters (EEG1, EEG2, EEG3/EMG)	N/A
_ChooseLowpassForEEG	Static	Asks user for lowpass value for a given EEG	eeg	String describing the current EEG (EEG1, EEG2, EEG3/EMG)	Integer number between 11-500 Hz for EEG	User input must be an integer between 11-500
_ValidateParams	Instance	Prints the parameter table and asks user if it is correct. User can edit table if it is incorrect.	N/A	N/A	N/A	N/A

_EditParams	Instance	Asks the user which device to edit, then asks for the new parameters	N/A	N/A	N/A	N/A
_SelectPODdeviceFromDictToEdit	Instance	Asks user for the pod device number to edit	N/A	N/A	Integer number for the pod device key	User must input an integer number that exists in the POD parameters dict keys
_GetForbiddenNames	Instance	Gets a list of all port names used by the POD devices. One string name can be excluded.	exclude=None	String of the name to exclude in the returned list of names	List of strings containing the port names for all POD devices	N/A
_PrintDeviceNumber	Static	Prints a string wrapper containing the device number	num	Integer representing the device number	N/A	N/A
_PrintPODdeviceParamDict	Instance	Prints string of the POD parameters dictionary. This can be used as a parameter to SetupPODparameters()	N/A	N/A	N/A	N/A
_DisplayPODdeviceParameters	Instance	Prints a table containing the parameters for all POD devices	N/A	N/A	N/A	N/A
_BuildFileName	Static	Adds the device number to the file name with path	fileName	String of the file path and name with extension	String of the filename with added device number: path\filename_<DEVICE#>.ext	N/A
			devNum	Integer number of the device		
_PrintSaveFile	Instance	Prints the path/file name of the file that streaming data is saved to	N/A	N/A	N/A	N/A
_CheckFileExt	Static	Checks for valid file extension	f	file name or extension	True if extension is in goodExt list, False otherwise	N/A
			flsExt=True	Boolean flag that is true if f is an extension, false otherwise		
			goodExt=['.csv','.txt','.edf']	List of valid file extensions		
			printErr=True	Boolean flag that, when true, will print an error statement		
_GetFilePath	Static	Asks user for a path and filename to save streaming data to.	N/A	N/A	String of the file path, name, and extension.	Filename must end in .csv, .txt, or .edf
_GetFileName	Static	Asks the user for a filename	N/A	N/A	String of the file name and extension	Filename must end in .csv, .txt, or .edf
_OpenSaveFile	Instance	Opens a file using the user's "path/filename_<DEVICE#>.extension"	devNum	Integer number for the DEVICE#	Opened write file	N/A
_OpenSaveFile_TXT	Static	Opens a text file and write the column names	fname	String filename	Opened file	N/A
_OpenSaveFile_EDF	Instance	Opens EDF file and write header	fname	String filename	Opened file	N/A
			devNum	Integer device number		
_WriteDataToFile_TXT	Static	Writes data to an open text file	file	opened write file	N/A	N/A
			data	List of 3 items, one for each channel		
			sampleRate	Integer sample rate in Hz		
			t	integer time (in seconds) corresponding to the data		
_WriteDataToFile_EDF	Static	Writes data to an open EDF file	file	opened EDF file	N/A	N/A
			data	List of 3 items, one for each channel		
_AskToStopStream	Instance	Writes command to stop streaming to all POD devices after user clicks enter	N/A	N/A	N/A	N/A
_StreamUntilStop	Static	Continually reads and translates streaming data from POD device until end stream command is read. Streaming data is written to a file every second.	pod	POD_8206HR device to read data from	N/A	N/A
			file	Open file to write data to		
			sampleRate	Sample rate for the pod device		
_StreamThreading	Instance	Creates a thread for each POD device to stream and a thread for user input to stop streaming.	N/A	N/A	N/A	N/A
_Stream	Instance	Tests that all devices are connected then starts streaming. It also prints the streaming execution time	N/A	N/A	N/A	N/A
_PrintOptions	Instance	Prints options available for user	N/A	N/A	N/A	N/A
_AskOption	Instance	Asks user which option to do	N/A	N/A	Integer number representing an option key	User input must be an integer that is a key in the options dictionary.
_DoOption	Instance	Performs the methods associated with the user selected option	choice	Integer number representing an option key	N/A	N/A
_AskYN	Static	Asks a user a yes or no question.	question	String of the question to ask the user	True when user inputs yes, False when user inputs no	User must input y/yes or n/no (case insentive)
_TimeFunc	Static	Runs a function and gets the calculated execultion time	func	function/method name	Float of the execution time in seconds rounded to 3 decimal places	N/A
_TestDeviceConnection	Static	Writes a PING command to the POD device and reads the response.	pod	POD_8206HR device to read data from	True if PING command is read from the device	N/A
_TestDeviceConnection_All	Instance	Checks the connection of all current POD devices	N/A	N/A	True if all devices are connected, false otherwise	N/A
_PrintInitCode	Instance	Prints code that can be used to initialize and run Setup_8206HR with the current parameters.	N/A	N/A	N/A	N/A
_uV	Static	Converts volts to microVolts, rounded to 6 decimal places	voltage	number of volts	number of uV	N/A

Class						
Name	**File**	**Description**	**Parent**	**Child**	**Author**	
POD_8206HR	PodDevice_8206HR.py	Handles communication using an 8206HR POD device.	POD_Basics	N/A	Thresa Kelly	
Imports						
Name	**Origin**	**Description**	**From**			
POD_Basics	Local	For inheritance	BasicPodProtocol			
POD_Packets	Local	For handling POD packets	PodPacketHandling			
POD_Commands	Local	For command constants	PodCommands			
Variables						
Name	**Scope**	**Description**	**Value**			
__B4LENGTH	Class	Constant containing the number of bytes for a full Binary4 packet	16			
__B4BINARYLENGTH	Class	Constant containing the number of bytes for binary data in a Binary4 packet	8			
_preampGain	Instance	Preamplifier gain	10 or 100			
Methods						
Name	**Type**	**Description**	**Parameter Name**	**Parameter Purpose**	**Return**	**Exception**
__init__	Dunder	Runs when an instance is constructed. It runs the parent's initialization. Then it updates the _commands to contain the appropriate commands for an 8306HR POD device.	port	String of the serial port to be opened. Used when initializing the COM_io instance.	N/A	N/A
			baudrate=9600	Integer baud rate of the opened serial port. Used when initializing the COM_io instance.		
UnpackPODpacket_Binary	Static	Overwrites the parent's method. Separates the components of a binary4 packet into a dictionary.	msg	Bytes string containing a complete binary4 Pod packet: STX (1 byte) + command (4 bytes) + packet number (1 bytes) + TTL (1 byte) + ch0 (2 bytes) + ch1 (2 bytes) + ch2 (2 bytes) + checksum (2 bytes) + ETX (1 byte)	A dictionary containing 'Command Number', 'Packet #', 'TTL', 'Ch0', 'Ch1', and 'Ch2' in bytes.	An exception is raised if (1) the packet does not have the minimum number of bytes, (2) does not begin with STX, or (3) does not end with ETX.
TranslatePODpacket_Binary	Static	Overwrites the parent's method. Unpacks the binary4 POD packet and converts the values of the ASCII-encoded bytes into integer values and the values of binary-encoded bytes into integers. Channel values are given in Volts.	msg	Bytes string containing a complete binary4 Pod packet: STX (1 byte) + command (4 bytes) + packet number (1 bytes) + TTL (1 byte) + ch0 (2 bytes) + ch1 (2 bytes) + ch2 (2 bytes) + checksum (2 bytes) + ETX (1 byte)	A dictionary containing 'Command Number', 'Packet #', 'TTL', 'Ch0', 'Ch1', and 'Ch2' as numbers.	N/A
TranslatePODpacket	Instance	Overwrites the parent's method. Determines if the packet is standard or binary, and translates accordingly. Adds a check for the 'GET TTL PORT' command.		Bytes string containing either a standard or binary packet	A dictionary containing the unpacked message in numbers	N/A
_TranslateTTLbyte_ASCII	Static	Separates the bits of each TTL (0-3) from a byte.	ttlByte	One Byte string for the TTL (ASCII encoded)	Tuple of the integer TTLs (0-3). 1 when input, 0 when output.	N/A
_TranslateTTLbyte_Binary	Static	Separates the bits of each TTL (0-3) from a byte.	ttlByte	One Byte string for the TTL (binary encoded)	Tuple of the integer TTLs (0-3). 1 when input, 0 when output.	N/A
_BinaryBytesToVoltage	Instance	Converts a binary bytes value read from POD device and converts it to the real voltage value at the preamplifier input	value	Bytes string containing voltage measurement	A number containing the voltage in Volts [V].	N/A
_Read_Binary	Instance	After receiving the prePacket, it reads the 8 bytes(TTL+channels) and then reads to ETX (checksum+ETX).	prePacket	Bytes string containing the beginning of a POD packet: STX (1 byte) + command number (4 bytes)	Byte string for a binary4 POD packet.	N/A
			validateChecksum=True	Set to True to validate the checksum. Set to False to skip validation		

Class						
Name	**File**	**Description**	**Parent**	**Child**	**Author**	
POD_8401HR	PodDevice_8401HR.py	Handles communication using an 8401HR POD device.	POD_Basics	N/A	Thresa Kelly	
Imports						
Name	**Origin**	**Description**	**From**			
POD_Basics	Local	For inheritance	BasicPodProtocol			
POD_Packets	Local	For handling POD packets	PodPacketHandling			
POD_Commands	Local	For command constants	PodCommands			
Variables						
Name	**Scope**	**Description**	**Value**			
__B5LENGTH	Class	number of bytes for a Binary 5 packet	31			
__B5BINARYLENGTH	Class	number of binary bytes for a Binary 5 packet	23			
_channelMap	Instance	Dictionary of the channel lables	Set by _GetChannelMapping(). Dictionary keys are ['A','B','C','D']			
_ssGain	Instance	Dictionary of the second stage gain for all four channels	1, 5, or None. Dictionary keys are ['A','B','C','D']			
_preampGain	Instance	Dictionary of the preamplifier gain for all four channels.	10, 100, or None. Dictionary keys are ['A','B','C','D']			
Methods						
Name	**Type**	**Description**	**Parameter Name**	**Parameter Purpose**	**Return**	**Exception**
__init__	Dunder	Runs when an instance is constructed. It runs the parent's initialization. Then it updates the _commands to contain the appropriate commands for an 8401HR POD device. Sets the _channelMap, _ssGain, and _preampGain.	port	String of the serial port to be opened. Used when initializing the COM_io instance.	N/A	An exception is raised if (1) the ssGain or preampGain have improper keys, (2) the device/sensor does not exist, (3) the ssGain was given bad values, or (4) the preampGain was given bad values
			deviceName	String of the corresponding device/sensor name		
			ssGain={'A':None,'B':None,'C':None,'D':None}	Dictionary of the secondary stage gain		
			preampGain={'A':None,'B':None,'C':None,'D':None}	Dictionary of the preamplifier gain		
			baudrate=9600	Integer baud rate of the opened serial port. Used when initializing the COM_io instance.		
UnpackPODpacket_Binary	Static	Overwrites the parent's method. Separates the components of a binary5 packet into a dictionary.	msg	Bytes string containing a complete binary5 Pod packet: STX (1 byte) + command (4) + packet number (1) + status (1) + channels (9) + analog inputs (12) + checksum (2) + ETX (1)	A dictionary containing 'Command Number', 'Packet #', 'Status', 'Channels', 'Analog EXT0', 'Analog EXT1', 'Analog TTL1', 'Analog TTL2', 'Analog TTL3', 'Analog TTL4', in bytes.	An exception is raised if (1) the packet does not have the minimum number of bytes, (2) does not begin with STX, or (3) does not end with ETX.
TranslatePODpacket_Binary	Instance	Overwrites the parent's method. Unpacks the binary5 POD packet and converts the values of the ASCII-encoded bytes into integer values and the values of binary-encoded bytes into integers. The channels and analogs are converted to volts (V).	msg	Bytes string containing a complete binary5 Pod packet: STX (1 byte) + command (4) + packet number (1) + status (1) + channels (9) + analog inputs (12) + checksum (2) + ETX (1)	A dictionary containing 'Command Number', 'Packet #', 'Status', 'CH3', 'CH2', 'CH1', 'CH0', 'Analog EXT0', 'Analog EXT1', 'Analog TTL1', 'Analog TTL2', 'Analog TTL3', 'Analog TTL4', as numbers.	N/A
GetChannelMapping	Static	Get the channel mapping (channel labels for A,B,C,D) for a given device.	deviceName	String for the device/sensor name.	Dictionary with keys A,B,C,D with values of the channel names. Returns None if the device name does not exist.	N/A
_Voltage_PrimaryChannels	Static	Converts a value to a voltage for a primary channel.	value	Value to be converted to voltage	Number of the voltage in volts [V]. Returns value if no gain is given (no-connect).	N/A
			ssGain=None	Second stage gain		
			PreampGain=None	Preamplifier gain		
_Voltage_PrimaryChannels_EEGEMG	Static	Converts a value to a voltage for an EEG/EMG primary channel.	value	Value to be converted to voltage	Number of the voltage in volts [V].	N/A
			ssGain	Second stage gain		
			PreampGain	Preamplifier gain		
_Voltage_PrimaryChannels_Biosensor	Static	Converts a value to a voltage for a biosensor primary channel.	value	Value to be converted to voltage	Number of the voltage in volts [V].	N/A
			ssGain	Second stage gain		
_Voltage_SecondaryChannels	Static	Converts a value to a voltage for a secondary channel.	value	Value to be converted to voltage	Number of the voltage in volts [V].	N/A
_Read_Binary	Instance	After receiving the prePacket, it reads the 23 bytes (binary data) and then reads to ETX (checksum+ETX).	prePacket	Bytes string containing the beginning of a POD packet: STX (1 byte) + command number (4 bytes)	Byte string for a binary5 POD packet.	N/A
			validateChecksum=True	Set to True to validate the checksum. Set to False to skip validation		

Class						
Name	**File**	**Description**	**Parent**	**Child**	**Author**	
POD_Packets	PodPacket Handling.py	Collection of methods for creating and interpreting POD packets	N/A	N/A	Thresa Kelly	
Imports						
Name	**Origin**	**Description**	**From**			
N/A	N/A	N/A	N/A			
Methods						
Name	**Type**	**Description**	**Parameter Name**	**Parameter Purpose**	**Return**	**Exception**
STX	Static	Get STX in bytes. STX marks the starting byte of a POD Packet	N/A	N/A	Bytes for STX (0x02)	N/A
ETX	Static	Get ETXin bytes. ETX marks the end byte of a POD Packet	N/A	N/A	Bytes for ETX(0x03)	N/A
IntToAsciiBytes	Static	Converts an integer value into ASCII-encoded bytes. First, it converts the integer value into a usable uppercase hexadecimal string. Then it converts the ASCII code for each character into bytes. Lastly, it ensures that the final message is the desired length. Example: if value=2 and numBytes=4, the returned ASCII will show b'0002', which is '0x30 0x30 0x30 0x32' in bytes.	value	Integer value to be converted into ASCII-encoded bytes	Bytes that are ASCII-encoded conversions of the value parameter.	N/A
			numBytes	Number bytes to be the length of the ASCII-encoded message.		
AsciiBytesToInt	Static	Converts a ASCII-encoded bytes message into an integer. It does this using a base-16 conversion.	msg_b	Bytes message to be converted to an integer. The bytes must be base-16 or the conversion will fail.	Integer result from the ASCII-encoded byte conversion.	N/A
BinaryBytesToInt	Static	Converts binary-encoded bytes into an integer	msg	Bytes message holding binary information to be converted into an integer.	Integer result from the binary-encoded bytes message.	N/A
			byteorder='big'	Ordering of bytes. 'big' for big endian and 'little' for little endian.		
			signed=False	Boolean flag to mark if the msg is signed (True) or unsigned (False)		
ASCIIbytesToInt_Split	Static	Converts a specific bit range in an ASCII-encoded bytes object to an integer.	msg	Bytes message holding binary information to be converted into an integer.	Integer result from the ASCII-encoded bytes message in a given bit range.	N/A
			keepTopBits	Integer position of the msb of desired bit range		
			cutBottomBits	Integer number of lsb to remove		
BinaryBytesToInt_Split	Static	Converts a specific bit range in a binary-encoded bytes object to an integer	msg	Bytes message holding binary information to be converted into an integer.	Integer result from the binary-encoded bytes message in a given bit range.	N/A
			keepTopBits	Integer position of the msb of desired bit range		
			cutBottomBits	Integer number of lsb to remove		
			byteorder='big'	Ordering of bytes. 'big' for big endian and 'little' for little endian.		
			signed=False	Boolean flag to mark if the msg is signed (True) or unsigned (False)		
Checksum		Calculates the checksum of a given bytes message. This is achieved by summing each byte in the message, inverting, and taking the last byte.	bytesIn	Bytes message containing POD packet data	Two ASCII-encoded bytes containing the checksum for bytesIn	N/A
BuildPODpacket_Standard		Builds a standard POD packet -- STX (1 byte) + command number (4 bytes) + optional packet (? bytes) + checksum (2 bytes) + ETX (1 bytes) -- as bytes.	commandNumber	Integer representing the command number. This will be converted into a 4 byte long ASCII-encoded bytes string.	Bytes string of a complete standard POD packet	N/A
			payload=None	bytes string containing the payload		
PayloadToBytes	Static	Converts a payload into a bytes string	payload	Integer, bytes, or tuple containing the payload	Bytes string of the payload	Raises an Exception when the payload argument is an incorrect type or formatted incorrectly.
			argSizes	Tuple of the argument sizes		

Class						
Name	**File**	**Description**	**Parent**	**Child**	**Author**	
POD_Basics	BasicPodProtocol.py	Handle basic communication with a POD device, including reading and writing packets and packet interpretation.	N/A	POD_8206HR	Thresa Kelly	
Imports						
Name	**Origin**	**Description**	**From**			
COM_io	Local	For opening and connecting serial COM ports	SerialCommunication			
POD_Packets	Local	For handling POD packets	PodPacketHandling			
POD_Commands	Local	Used to contain all POD commands in the class instance	PodCommands			
Variables						
Name	**Scope**	**Description**	**Value**			
__numPod	Class	Integer equal to the number of POD_Basics class instances. Incremented on construction and decremented on destruction	0			
__MINSTANDARDLENGTH	Class	integer minimum number of bytes in a standard POD packet	8			
__MINBINARYLENGTH	Class	integer minimum number of bytes in a binary POD packet	15			
_port	Instance	Open serial port via COM_io class instance	COM_io			
_commands	Instance	Command handler POD_Commands class instance	POD_Commands			
Methods						
Name	**Type**	**Description**	**Parameter Name**	**Parameter Purpose**	**Return**	**Exception**
__init__	Dunder	Runs when an instance of POD_Basics is constructed. It initializes the instance variable for the COM port communication (_port) and for the command handler (_commands). It also increments the POD device counter (__NUMPOD).	port	String of the serial port to be opened. Used when initializing the COM_io instance.	N/A	N/A
			baudrate=9600	Integer baud rate of the opened serial port. Used when initializing the COM_io instance.		
__del__	Dunder	Runs when an instance is destructed. It decrements the POD device counter (__NUMPOD)	N/A	N/A	N/A	N/A
GetNumberOfPODDevices	Static	Get the POD device counter	N/A	N/A	Integer of the number of class instances (__NUMPOD).	N/A
UnpackPODpacket_Standard	Static	Converts a standard POD packet into a dictionary containing the command number and payload (if applicable) in bytes.	msg	Bytes message containing a standard POD packet: STX (1 byte) + command number (4 bytes) + optional packet (? bytes) + checksum (2 bytes) + ETX (1 bytes)	A dictionary containing the POD packet's 'Command Number' and 'Payload' (if applicable) in bytes.	An exception is raised if (1) the msg does not have the minimum number of bytes in a standard pod packet, (2) does not begin with STX, and (3) does not end with ETX.
UnpackPODpacket_Binary	Static	Converts a variable-length binary packet into a dictionary containing the command number, binary packet length, and binary data in bytes.	msg	Bytes message containing a variable-length POD packet: STX (1 byte) + command number (4 bytes) + length of binary (4 bytes) + checksum (2 bytes) + ETX (1 bytes) '+ binary (LENGTH bytes) + checksum (2 bytes) + ETX (1 bytes)	A dictionary containing the 'Command Number', 'Binary Packet Length', and 'Binary Data' in bytes.	An exception is raised if (1) the msg does not have the minimum number of bytes in a standard pod packet, (2) does not begin with STX, (3) does not end with ETX., and (4) does not have an ETX after standard packet.
TranslatePODpacket_Standard	Instance	Unpacks the standard POD packet and converts the ASCII-encoded bytes values into integer values.	msg	Bytes message containing a standard POD packet	A dictionary containing the POD packet's 'Command Number' and 'Payload' (if applicable) in integers.	N/A
TranslatePODpacket_Binary	Static	Unpacks the variable-length binary POD packet and converts the values of the ASCII-encoded bytes into integer values and leaves the binary-encoded bytes as is.	msg	Bytes message containing a variable-length POD packet	A dictionary containing the 'Command Number' and 'Binary Packet Length' in integers, and 'Binary Data' in bytes.	N/A
_ValidateChecksum	Static	Validates the checksum of a given POD packet. The checksum is valid if the calculated checksum from the data matches the checksum written in the packet.	msg	Bytes message containing a POD packet: STX (1 bytes) + data (? bytes) + checksum (2 bytes) + ETX (1 byte).	Returns True if the checksum is correct, false otherwise.	An exception is raised if the msg does not begin with STX or end with ETX.
GetDeviceCommands	Instance	Gets the dictionary containing the class instance's available POD commands.	N/A	N/A	Dictionary containing the available commands and their information. Formatted as key(command number) : value([command name, number of argument ASCII bytes, number of return bytes, binary flag])	N/A
SetBaudrateOfDevice	Instance	If the port is open, it will change the baud rate to the parameter's value	baudrate	Integer baud rate to set for the open serial port.	True if successful at setting the baud rate, false otherwise	N/A
UnpackPODpacket	Static	Determines if the packet is standard or binary, and unpacks accordingly.	msg	Bytes string containing either a standard or binary packet	A dictionary containing the unpacked message in bytes	N/A
TranslatePODpacket	Instance	Determines if the packet is standard or binary, and translates accordingly.	msg	Bytes string containing either a standard or binary packet	A dictionary containing the unpacked message in numbers	N/A
WriteRead	Instance	Writes a command with optional payload to POD device, then reads (once) the device response.	cmd	An integer representing the command number.	Bytes string containing a POD packet beginning with STX and ending with ETX. This may be a standard packet, binary packet, or an unformatted packet (STX+something+ETX).	N/A
			payload=None	None when there is no payload. If there is a payload, set to an integer value or a bytes string.		
			validateChecksum=True	Set to True to validate the checksum. Set to False to skip validation		
		Builds a POD packet and writes it to a POD device via	cmd	An integer representing the command number.		An exception is raised if (1) the command does not

GetPODpacket	Instance	Builds a POD packet and writes it to a POD device via COM port. If an integer payload is give, the method will convert it into a bytes string of the length expected by the command. If a bytes payload is given, it must be the correct length.	payload=None	None when there is no payload. If there is a payload, set to an integer value, bytes string, or tuple	Returns the bytes string of the POD packet.	exist for the instance, (2) a payload is not given when the command expects one, (3) the payload (given in bytes) is the size not expected by the command, or (4) the payload is given as a type other than integer or bytes.
WritePacket	Instance	Builds a POD packet and writes it to the POD device.	cmd	An integer representing the command number.	Returns the bytes string that was written to the POD device	N/A
			payload=None	None when there is no payload. If there is a payload, set to an integer value, bytes string, or tuple		
ReadPODpacket	Instance	Reads a complete POD packet, either in standard or binary format, beginning with STX and ending with ETX. Reads first STX and then starts recursion.	validateChecksum=True	Set to True to validate the checksum. Set to False to skip validation	Bytes string containing a POD packet beginning with STX and ending with ETX. This may be a standard packet, binary packet, or an unformatted packet (STX+something+ETX).	N/A
_ReadPODpacket_Recursive	Instance	Reads the command number. If the command number ends in ETX, the packet is returned. Next, it checks if the command is allowed. Then, it checks if the command is standard or binary and reads accordingly, then returns the packet.	validateChecksum=True	Set to True to validate the checksum. Set to False to skip validation	Bytes string containing a POD packet beginning with STX and ending with ETX. This may be a standard packet, binary packet, or an unformatted packet (STX+something+ETX).	N/A
_Read_GetCommand	Instance	Reads one byte at a time up to 4 bytes to get the ASCII-encoded bytes command number. For each byte read, it can (1) start the recursion over if an STX is found, (2) returns if ETX is found, or (3) continue building the command number.	validateChecksum=True	Set to True to validate the checksum. Set to False to skip validation	4 byte long string containing the ASCII-encoded command number.	An exception is raised if the command number is not allowed for the POD device
_Read_ToETX	Instance	Reads one byte at a time until an ETX is found. It will restart the recursive read if an STX is found anywhere.	validateChecksum=True	Set to True to validate the checksum. Set to False to skip validation	Bytes string ending with ETX	N/A
_Read_Standard	Instance	Reads the payload, checksum, and ETX. Then it builds the complete standard POD packet in bytes.	prePacket	Bytes string containing the beginning of a POD packet: STX (1 byte) + command number (4 bytes)	Bytes string for a complete standard POD packet	An exception is raised if the checksum is invalid (only if validateChecksum=True)
			validateChecksum=True	Set to True to validate the checksum. Set to False to skip validation		
_Read_Binary	Instance	Reads the remaining part of the variable-length binary packet. It first reads the standard packet (prePacket+payload+checksum+ETX). Then it determines how long the binary packet is from the payload of the standard POD packet and reads that many bytes. It then reads to ETX to get the checksum+ETX.	prePacket	Bytes string containing the beginning of a POD packet: STX (1 byte) + command number (4 bytes)	Bytes string for a variable-length binary POD packet	An exception is raised if the checksum is invalid (only if validateChecksum=True)
			validateChecksum=True	Set to True to validate the checksum. Set to False to skip validation		

Class						
Name	**File**	**Description**	**Parent**	**Child**	**Author**	
POD_Commands	PodCommands.py	Manages a dictionary containing available commands for a POD device.	N/A	N/A	Thresa Kelly	
Imports						
Name	**Origin**	**Description**	**From**			
N/A	N/A	N/A	N/A			
Variables						
Name	**Scope**	**Description**	**Value**			
__NAME	Class	index key for the command name for __commands list values	0			
__ARGUMENTS	Class	index key for the number of bytes in an argument for __commands list values	1			
__RETURNS	Class	index key for the number of bytes in the return for __commands list values	2			
__BINARY	Class	index key for the binary flag for __commands list values	3			
__NOVALUE	Class	Integer used to mark when a list item in __commands means 'no value' or undefined.	-1			
__U8	Class	Number of bytes for an unsigned 8-bit value	2			
__U16	Class	Number of bytes for an unsigned 16-bit value	4			
__commands	Instance	Dictionary containing the available commands for a POD device. Each entry is formatted as { key(command number) : value([command name, number of argument ASCII bytes, number of return bytes, binary flag)) }	{ key : value }			
Methods						
Name	**Type**	**Description**	**Parameter Name**	**Parameter Purpose**	**Return**	**Exception**
__init__	Dunder	Runs whan an instance is constructed. It sents the commands dictionary to the basic command set.	N/A	N/A	N/A	N/A
NoValue	Static	Gets value of __NOVALUE	N/A	N/A	Value of __NOVALUE	N/A
U8	Static	Gets value of __U8	N/A	N/A	Value of __U8	N/A
U16	Static	Gets value of __U16	N/A	N/A	Value of __U16	N/A
GetBasicCommands	Static	Creates a dictionary containing the basic POD command set (0,1,2,3,4,5,6,7,8,9,10,11,12)	N/A	N/A	N/A	N/A
GetCommands	Instance	Gets the contents of the current command dictionary (__commands)	N/A	N/A	N/A	N/A
RestoreBasicCommands	Instance	Sets the current commands (__commands) to the basic POD command set.	N/A	N/A	N/A	N/A
AddCommand	Instance	Adds a command entry to the current commands dictionary (__commands) if the command does not exist	commandNumber	Integer of the command number	True if the command was successfully added, False if the command could not be added because t already exists.	N/A
			commandName	String of the command's name		
			argumentBytes	Integer of the number of bytes in the argument		
			returnBytes	Integer of the number of bytes in the return		
			isBinary	Boolean flag to mark if the command is binary (True) or standard (False)		
RemoveCommand	Instance	Removes the entry for a given command in __commands dictionary.	cmd	integer command number or string command name.	True if the command was successfully removed, False if the command does not exist.	N/A
CommandNumberFromName	Instance	Gets the command number from the command dictionary using the command's name	name	string of the command's name	Integer representing the command number. If the command could not be found, return None.	N/A
ArgumentBytes	Instance	Gets the tuple for the number of bytes in the argument for a given command.	cmd	integer command number or string command name.	Tuple representing the number of bytes in the argument for cmd. If the command could not be found, return None.	N/A
ReturnBytes	Instance	Gets the tuple for the number of bytes in the return for a given command.	cmd	integer command number or string command name.	Tuple representing the number of bytes in the return for cmd. If the command could not be found, return None.	N/A
IsCommandBinary	Instance	Gets the binary flag for a given command	cmd	integer command number or string command name.	Boolean flag that is True if the command is binary and False if standard. If the command could not be found, return None.	N/A
DoesCommandExist	Instance	Checks if a command exists in the __commands dictionary	cmd	integer command number or string command name.	True if the command exists, false otherwise.	N/A

Class						
Name	**File**	**Description**	**Parent**	**Child**	**Author**	
COM_io	SerialCommunication.py	Handle serial communication (read/write) using COM ports.	N/A	N/A	Thresa Kelly	
Imports						
Name	**Origin**	**Description**	**From**			
serial.tools.list_ports	Enviornment	For accessing the COM ports on the computer	N/A			
Variables						
Name	**Scope**	**Description**	**Value**			
__serialInst	Instance	Serial object to set the port and baud rate to. It can be opened or closed.	Serial			
Methods						
Name	**Type**	**Description**	**Parameter Name**	**Parameter Purpose**	**Return**	**Exception**
GetCOMportsList	Static	Finds all the available COM ports on the user's computer and appends them to an accessible list.	N/A	N/A	List containing the names of available COM ports	N/A
__init__	Dunder	Runs when the object is constructed. It initialized the __serialInst to a given COM port with a set baudrate.	port	String of the serial port to be opened.	N/A	N/A
			baudrate=9600	Integer baud rate of the opened serial port.		
__del__	Dunder	Runs when the object is destructed. It closes the serial port, if open.	N/A	N/A	N/A	N/A
__BuildPortName	Instance	Converts the port parameter into the "COM"+<number> format	port	String name of a COM port. Can be an integer or string.	N/A	N/A
IsSerialOpen	Instance	Returns True if the serial instance port is open, false otherwise	N/A	N/A	N/A	N/A
IsSerialClosed	Instance	Returns False if the serial instance port is open, True otherwise	N/A	N/A	N/A	N/A
CloseSerialPort	Instance	Closes the instance serial port if it is open.	N/A	N/A	N/A	N/A
OpenSerialPort	Instance	First, it closes the serial port if it is open. Then, it opens a serial port with a set baud rate.	port	String of the serial port to be opened.	N/A	Raises an exception if the given port does not exist.
			baudrate=9600	Integer baud rate of the opened serial port.		
SetBaudrate	Instance	If the port is open, it will change the baud rate to the parameter's value	baudrate	Integer baud rate to set for the open serial port.	True if successful at setting the baud rate, false otherwise	N/A
GetPortName	Instance	Gets the name of the open port.	N/A	N/A	If the serial port is open, it will return a string of the port's name. If the port is closed, it will return None.	N/A
Read	Instance	Reads a specified number of bytes from the open serial port.	numBytes	Integer number of bytes to read	If the serial port is open, it will return a set number of read bytes. If it is closed, it will return None.	N/A
ReadLine	Instance	Reads until a new line ('\n') from the open serial port.	N/A	N/A	If the serial port is open, it will return a complete read line. If closed, it will return None.	N/A
ReadUntil	Instance	Reads until a set character from the open serial port.	eol	end-of-line character	If the serial port is open, it will return a read line ending in eol. If closed, it will return None.	N/A
Write	Instance	Write a set message to the open serial port.	message	byte string containing the message to write	N/A	N/A