

---

# **githex Documentation**

***Release 1.0***

**William Cember**

March 07, 2015



<b>1</b>	<b>Bithex</b>	<b>3</b>
1.1	Installation . . . . .	3
1.2	Quickstart . . . . .	3
1.3	Copyright and License . . . . .	3
<b>2</b>	<b>Developer Interface</b>	<b>5</b>
2.1	Functions . . . . .	5
2.2	Exceptions . . . . .	6
<b>3</b>	<b>Indices and tables</b>	<b>7</b>



Contents:



Bithex is a Python 2 library to analyze bitcoin scripts.

## 1.1 Installation

The current release of bithex is available through pip:

```
$ pip install bithex
```

## 1.2 Quickstart

```
>>> from bithex import compile_hex
>>> compile_hex('aa206fe28c0ab6f1b372c1a6a246ae63f74f931e8365e15a089c68d61900000000087')
'OP_HASH256 6fe28c0ab6f1b372c1a6a246ae63f74f931e8365e15a089c68d6190000000000 OP_EQUAL'
```

## 1.3 Copyright and License

Copyright (c) 2015 William Cember

Licensed under the MIT License

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.





---

## Developer Interface

---

### 2.1 Functions

`bithex.compile_hex(hex_string)`

Compile a hex string into Script.

**Parameters** `hex_string` – A string or unicode string that is the hex representation of a bitcoin script.

**Returns** A bitcoin script formatted as a string.

**Return type** `str`

**Raises**

- `InvalidHexError` – Raised if the input `hex_string` doesn't compile to valid Script.
- `TypeError` – Raised if the input `hex_string` isn't a string.

#### Examples

```
>>> compile_hex('aa106fe28c0ab6f1b372c1a6a246ae63f74f87')
'OP_HASH256 6fe28c0ab6f1b372c1a6a246ae63f74f OP_EQUAL'
>>> compile_hex('76a90a134408afa258a50ed7a188ac')
'OP_DUP OP_HASH160 134408afa258a50ed7a1 OP_EQUALVERIFY OP_CHECKSIG'
>>> compile_hex('0504b0bd6342ac')
'04b0bd6342 OP_CHECKSIG'
>>> compile_hex('aa206fe28c0ab6f1b372c1a6a246ae63f74f931')
InvalidHexError
```

`bithex.classify_hex(hex_string)`

Classify the transaction type of a hex string.

**Parameters** `hex_string` – A string or unicode string that is the hex representation of a bitcoin script.

**Returns** The transaction type of the hex string per <https://bitcoin.org/en/developer-guide#standard-transactions>.

**Return type** `str`

**Raises**

- `TypeError` – Raised if the input `hex_string` isn't a string.
- `InvalidHexError` – Raised if the input `hex_string` doesn't compile to valid Script.

### Examples

```
>>> classify_hex('76a914a134408afa258a50ed7a1d9817f26b63cc9002cc88ac')
'P2PKH'
>>> classify_hex('aa106fe28c0ab6f1b372c1a6a246ae63f74f87')
'nonstandard transaction'
```

`bithex.classify_script(script)`

Classify the transaction type of a script.

**Parameters** `script` – A bitcoin script.

**Returns** The transaction type of the hex string per <https://bitcoin.org/en/developer-guide#standard-transactions>. The following are the values that can be returned: 'P2PKH', 'P2SH', 'Multisig', 'Pubkey', 'Null Data', 'nonstandard transaction'.

**Return type** `str`

**Raises** `TypeError` – Raised if the input script isn't a string.

### Examples

```
>>> classify_script('OP_DUP OP_HASH160 a13 OP_EQUALVERIFY OP_CHECKSIG')
'P2PKH'
>>> classify_script('OP_HASH160 54c557e07dde5bb6cb791c7a540e OP_EQUAL')
'P2SH'
>>> classify_script('OP_HASH256 6fe28c0ab6f1b372c1a6a246ae63 OP_EQUAL')
'nonstandard transaction'
```

## 2.2 Exceptions

**exception** `bithex.InvalidHexError(value)`

Raised by `compile_hex`, `classify_hex` if the input doesn't compile to valid Script.

---

## Indices and tables

---

- *genindex*
- *modindex*
- *search*



## C

`classify_hex()` (in module `bithex`), [5](#)  
`classify_script()` (in module `bithex`), [6](#)  
`compile_hex()` (in module `bithex`), [5](#)

## I

`InvalidHexError`, [6](#)