Basic GET

GET a single resource via its URI

Default operation is a GET:

```
curl http://api.example.com:8080/statuses/1234
```

GET multiple resources where IDs are in a range

Use square brackets with a dashed range:

```
curl http://api.example.com:8080/items/[1230-1234]
```

GET multiple resources where IDs aren't in a range

Use curly braces with comma-delimited strings:

```
curl http://api.example.com:8080/products/{abc,def,ghi}/status
```

Using HTTP Headers

Accept only the application/json content-type

Use the header option: `-H` or `--header`

```
curl -H 'Accept: application/json' http://api.example.com:8080/items/1234
```

Add multiple headers

Use multiple `-H` options:

```
curl -H 'Accept: application/json' -H 'Accept-Encoding: gzip' http://api.example.com:8080/products/a1b2c3ef/status
```

Note: the output from this is likely to be unreadable, because it's gzipped!

More likely you'd use this with ETags:

```
curl -H 'Accept: application/json' -H 'If-None-Match: "1cc044-172-3d9aee80"'
    http://api.example.com:8080/products/a1b2c3ef/status
```

Show the network and HTTP "conversation"

Use the verbose option: -v or --verbose

```
curl -v -H http://api.example.com
```

POST and PUT

POST data to a URI

To send data to the server, you use either POST or PUT, depending on what the API requires. To do a POST, you simply use the `-d` (`--data`) with some content:

```
curl -d "name=Ted" http://api.example.com:8080/customers
```

Note that this uses `application/x-www-form-urlencoded` as the Content-Type, i.e., as if it was submitted by an HTML Form. You can use multiple `-d` options, which will be combined, e.g., these two commands produce the same content:

```
curl -d "first=Ted" -d "last=Young" http://api.example.com:8080/customers
```

```
curl -d "first=Ted&last=Young" http://api.example.com:8080/customers
```

If you want to send JSON, you'll need to specify the Content-Type explicitly using the `-H` (header) option:

```
curl -d '{"name": "Ted"}' -H 'Content-Type: application/json' http://api.example.com:8080/items
```

PUT data to a URI

If you need to use the PUT method, you'll need to override the method with the `-X` (`--request`) option:

```
curl -X PUT -d '{"name": "Ted"}' -H 'Content-Type: application/json' http://api.example.com:8080/items/1234
```