Newman CLI Cheatsheet

A handy reference for using Newman, the command-line collection runner for Postman, to execute and test API collections efficiently.

Basic Execution

Running Collections

`newman run <collection-file>` - Executes a Postman collection from a file.

Example:

```
newman run my_collection.json
```

`newman run <collection-url>` - Executes a Postman collection directly from a URL.

Example:

```
newman run https://example.com/my_collection.json
```

`-e, --environment <environment-file>` - Specifies an environment file to use during the collection run.

Example:

```
newman run my_collection.json -e dev_environment.json
```

`--globals <globals-file>` - Specifies a globals file to use during the collection run. Globals override environment variables.

Example:

```
newman run my_collection.json --globals globals.json
```

`--iteration-data <data-file>` - Specifies a data file (JSON or CSV) to use for multiple iterations.

Example:

```
newman run my_collection.json --iteration-data data.csv
```

`--reporters <reporter-name>` - Specifies the reporters to use during the collection run.

Example:

```
newman run my_collection.json --reporters cli,json
```

Output and Reporting

`--reporter-<reporter-name>-<option> <value>` - Configures reporter-specific options.

Example:

```
newman run my_collection.json --reporters json --reporter-json-export results.json
```

`--export-environment <file>` - Exports the resolved environment variables to a file after the run.

Example:

```
newman run my_collection.json --export-environment exported_env.json
```

`--export-globals <file>` - Exports the resolved global variables to a file after the run.

Example:

```
newman run my_collection.json --export-globals exported_globals.json
```

`--verbose` - Shows verbose output for debugging purposes.

Example:

```
newman run my_collection.json --verbose
```

Advanced Options

Controlling Execution Flow

`--delay <milliseconds>` - Adds a delay (in milliseconds) between request iterations.

Example:

```
newman run my_collection.json --delay 500
```

`--bail` - Stops the collection run on the first error.

Example:

```
newman run my_collection.json --bail
```

`--no-bail` - Continues the collection run even when there are script errors.

Example:

```
newman run my_collection.json --no-bail
```

`--folder <folder-name>` - Runs only a specific folder within the collection.

Example:

```
newman run my_collection.json --folder "Users API"
```

`--test-report-html <path>` - Generates a HTML test report.

Example:

```
newman run collection.json --test-report-html report.html
```

Request Configuration

`--request-timeout <ms>` - Sets the request timeout in milliseconds.

Example:

```
newman run my_collection.json --request-timeout 5000
```

`--ignore-redirects` - Prevents Newman from automatically following HTTP redirects.

Example:

```
newman run my_collection.json --ignore-redirects
```

`--ssl-client-cert <file>` - Specifies the path to the SSL client certificate file.

Example:

```
newman run my_collection.json --ssl-client-cert client.crt
```

`--ssl-client-key <file>` - Specifies the path to the SSL client key file.

Example:

```
newman run my_collection.json --ssl-client-key client.key
```

Integration & Scripting

Using with CI/CD

Newman can be easily integrated into CI/CD pipelines using tools like Jenkins, Travis CI, or GitLab CI.
Use the appropriate commands in your CI/CD script to run Newman and generate reports.

Example (Jenkins):

```
newman run my_collection.json -e dev_environment.json --reporters cli,json
--reporter-json-export results.json
```

Fail the CI/CD build based on Newman's exit code. A non-zero exit code typically indicates test failures.

Pre-request & Test Scripts

Newman executes pre-request and test scripts defined within your Postman collections. These scripts allow you to dynamically set variables, perform assertions, and more.

Example (setting an environment variable):

```
pm.environment.set("variable_name", "variable_value");
```

Example (performing an assertion):

```
pm.test("Status code is 200", function () {
    pm.response.to.have.status(200);
});
```

Troubleshooting

Common Issues

Collection not found:
Ensure the collection file path or URL is correct and accessible.

Environment/Globals file not found:
Verify the file paths are accurate.

Request timeout:
Increase the `--request-timeout` value if requests are taking too long.

SSL errors:
Ensure your SSL configuration is correct, especially when using client certificates.

Script errors:
Check the console output for errors in your pre-request or test scripts. Use `--verbose` for more detailed debugging information.

Debugging Tips

Use the `--verbose` flag to get detailed logs of the collection run. This can help identify issues with requests, scripts, or environment variables.

Inspect the generated reports (JSON, HTML) to analyze test results and identify failing requests.

Use `console.log()` statements in your pre-request and test scripts to output debugging information to the console.

Validate your collection and environment files in Postman before running them with Newman.