Living Documentation

Table of Contents

Summary

Scenarios			Steps							Features: 22	
Passed	Failed	Total	Passed	Failed	Skipped	Pending	Undefined	Missing	Total	Duration	Status
Asynchronous failing steps											
1	1	2	5	0	2	0	1	0	8	019ms	failed
Asynchronous pending steps											
1	0	1	6	0	0	0	0	0	6	013ms	passed
Asynchronous step definition callbacks											
1	0	1	4	0	0	0	0	0	4	061ms	passed
Attachments											
7	0	7	35	0	0	0	0	0	35	01s 711ms	passed
Background											
2	0	2	16	0	0	0	0	0	16	026ms	passed
CoffeeScript support											
0	1	1	0	0	3	0	1	0	4	000ms	failed
Command line interface											
11	0	11	50	0	0	0	0	0	50	02s 464ms	passed
Core feature elements execution											
5	0	5	23	0	0	0	0	0	23	053ms	passed
Data Tables											
1	0	1	3	0	0	0	0	0	3	007ms	passed
Environment Hooks											
12	0	12	59	0	0	0	0	0	59	02s 906ms	passed
JSON Formatter											
26	0	26	104	0	0	0	0	0	104	07s 438ms	passed
PogoScript support											
0	1	1	0	0	3	0	1	0	4	000ms	failed
Pretty Formatter											
3	0	3	13	0	0	0	0	0	13	816ms	passed
Scenario Outlines and Examples											

Scenarios			Steps							Features: 22	
4	0	4	31	0	0	0	0	0	31	050ms	passed
Scenario Statuses											
6	0	6	30	0	0	0	0	0	30	01s 707ms	passed
Step definition callbacks											
1	3	4	4	0	6	0	6	0	16	010ms	failed
step definition snippets											
1	3	4	9	0	0	0	4	0	13	020ms	failed
step definitions with string pattern											
2	1	3	13	0	0	0	1	0	14	012ms	failed
Strict mode											
5	0	5	23	0	0	0	0	0	23	01s 366ms	passed
Summary Formatter											
2	0	2	8	0	0	0	0	0	8	514ms	passed
User logs into the system											
0	6	6	0	0	0	0	29	0	29	000ms	failed
World constructor callback with object											
1	0	1	4	0	0	0	0	0	4	006ms	passed
Totals											
92	16	108	440	0	14	0	43	0	497	19s 210ms	

Features

Asynchronous failing steps

Scenario: see asynchronously failing scenarios

Given

the following feature: 👍 (000ms)

```
Feature: a feature
  Scenario: a failing scenario
    When I divide 10 by 0
    Then the result is 9
```

And

the step "I divide 10 by 0" has a mapping asynchronously failing with the message "Divide by 0, uh?" 👍 (000ms)

When

Cucumber runs the feature 👍 (019ms)

Then

the scenario called "a failing scenario" is reported as failing 👍 (000ms)

Scenario: see asynchronously failing scenarios with exception

tags: @untestable-on-self

Given

the following feature: 👍 (000ms)

```
Feature: a feature
  Scenario: a failing scenario
    When I divide 10 by 0
    Then the result is 9
```

And

the step "I divide 10 by 0" has a mapping asynchronously failing through an exception with the message "Divide by 0, uh?" 👎 (000ms)

When

Cucumber runs the feature 👎 (000ms)

Then

the scenario called "a failing scenario" is reported as failing 👎 (000ms)

Asynchronous pending steps

Scenario: Pending step means the scenario is pending

Given

a scenario with: 👍 (000ms)

```
When I add 4 and 5
Then the result is 9
```

And

the step "I add 4 and 5" has an asynchronous pending mapping 👍 (000ms)

And

the step "the result is 9" has a passing mapping 👍 (000ms)

When

Cucumber executes the scenario 👍 (013ms)

Then

the scenario is pending 👍 (000ms)

And

the step "the result is 9" is skipped 👍 (000ms)

Asynchronous step definition callbacks

In order to test asynchronous code
As a dev
I want step definitions to call back asynchronously

Scenario: Execute feature with asynchronous step definition

Given

a step definition matching /^an asynchronous step passes$/ calling back asynchronously after 50 milliseconds 👍 (000ms)

And

a step definition matching /^a step passes$/ 👍 (000ms)

When

I run the following feature: 👍 (060ms)

```
Feature: Asynchronous step definition body
  Scenario: Waiting for an asynchronous step to call back
    When an asynchronous step passes
    Then a step passes
```

Then

the feature should have run successfully 👍 (000ms)

Attachments

Scenario: Attach a buffer

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.Before(function(scenario, callback) {
    scenario.attach(new Buffer([100, 97, 116, 97]), 'image/png');
    callback();
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (196ms)

Then

it outputs this json: 👍 (002ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Before ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {},
            "embeddings": [
              {
```

```
              "mime_type": "image/png",
              "data": "ZGF0YQ=="
            }
          ]
        },
        {
          "name": "This step is passing",
          "line": 4,
          "keyword": "Given ",
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {}
        }
      ]
    }
  ]
}
]
```

Scenario: Attach a stream

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.Before(function(scenario, callback) {
    var Stream = require('stream');
    var versionParts = /v(\d+)\.(\d+)\.(\d+)/.exec(process.version);
    var major = parseInt(versionParts[0], 10);
    var minor = parseInt(versionParts[1], 10);

    if (major > 0 || minor >= 10) {
      var stream = new Stream.Readable();
      stream._read = function() {};
      stream.push(new Buffer([100, 97, 116, 97]));
      stream.push(null);

      scenario.attach(stream, 'image/png', function(error) {
        callback(error);
      });
    }
    else {
      scenario.attach(new Buffer([100, 97, 116, 97]), 'image/png');
      callback();
    }
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (249ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
```

```
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Before ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {},
            "embeddings": [
              {
                "mime_type": "image/png",
                "data": "ZGF0YQ=="
              }
            ]
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```


Scenario: Attach from an around hook (pre scenario)

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.Around(function(scenario, runScenario) {
    scenario.attach("text");

    runScenario(function(scenario, callback) {
      callback();
    });
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (286ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
```

```
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {},
            "embeddings": [
              {
                "mime_type": "text/plain",
                "data": "dGV4dA=="
              }
            ]
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```


Scenario: Attach from an around hook (post scenario)

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.Around(function(scenario, runScenario) {
    runScenario(function(callback) {
      scenario.attach("text");
      callback();
    });
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (258ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
```

```
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {},
            "embeddings": [
              {
                "mime_type": "text/plain",
                "data": "dGV4dA=="
              }
            ]
          }
        ]
      }
    ]
```

```
  }
]
```

Scenario: Attach from a before hook

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.Before(function(scenario, callback) {
    scenario.attach("text");
    callback();
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (208ms)

Then

it outputs this json: 👍 (000ms)

```
[
```

```
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Before ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {},
            "embeddings": [
              {
                "mime_type": "text/plain",
                "data": "dGV4dA=="
              }
            ]
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: Attach from an after hook

Given

a file named "features/a.feature" with: 👍 (002ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.After(function(scenario, callback) {
    scenario.attach("text");
    callback();
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (247ms)

Then

it outputs this json: 👍 (002ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
```

```
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {},
            "embeddings": [
              {
                "mime_type": "text/plain",
                "data": "dGV4dA=="
              }
            ]
          }
        ]
      }
    ]
  }
]
```


Scenario: Attach from a step definition

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) {
    var world = this;
    world.scenario.attach("text");
    callback();
  });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.Before(function(scenario, callback) {
    var world = this;
    world.scenario = scenario;
    callback();
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (215ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
```

```
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Before ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {},
            "embeddings": [
              {
                "mime_type": "text/plain",
                "data": "dGV4dA=="
              }
            ]
          }
        ]
      }
    ]
  }
]
```

Background

Background allows you to add some context to the scenarios in a single feature. A Background is much like a scenario containing a number of steps. The difference is when it is run. The background is run before each of your scenarios but after any of your Before Hooks.

Scenario: One scenario and a background

Given

the following feature: 👍 (000ms)

```
Feature: testing scenarios
  Background:
    Given a background step

  Scenario:
    When a scenario step
```

And

the step "a background step" has a passing mapping 👍 (000ms)

And

the step "a scenario step" has a passing mapping 👍 (000ms)

When

Cucumber runs the feature 👍 (009ms)

Then

the feature passes 👍 (000ms)

And

the step "a background step" passes 👍 (000ms)

And

the step "a scenario step" passes 👍 (000ms)

Scenario: Two scenarios and a background

Given

the following feature: 👍 (000ms)

```
Feature: testing scenarios
  Background:
    Given a background step

  Scenario:
    When a scenario step

  Scenario:
    When a second scenario step
```

And

the step "a background step" has a passing mapping 👍 (000ms)

And

the step "a scenario step" has a passing mapping 👍 (000ms)

And

the step "a second scenario step" has a passing mapping 👍 (000ms)

When

Cucumber runs the feature 👍 (014ms)

Then

the feature passes 👍 (000ms)

And

the step "a background step" passes 👍 (000ms)

And

the step "a scenario step" passes 👍 (000ms)

And

the step "a second scenario step" passes 👍 (000ms)

CoffeeScript support

In order to use the JS dialect I’m most comfortable with
As a step definition implementor
I want to use CoffeeScript for writing step definitions

Scenario: CoffeeScript step definition

Given

a mapping written in CoffeeScript 👎 (000ms)

When

Cucumber executes a scenario using that mapping 👎 (000ms)

Then

the feature passes 👎 (000ms)

And

the mapping is run 👎 (000ms)

Command line interface

In order to run cucumber in different contexts
As a person who wants to run features
I want to run Cucumber on the command line

Scenario: run a single feature

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature
  Scenario:
    When a step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.When(/^a step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f progress features/a.feature` 👍 (297ms)

Then

it outputs this text: 👍 (000ms)

```
.

1 scenario (1 passed)
1 step (1 passed)
```

And

the exit status should be 0 👍 (000ms)

Scenario: run a single scenario within feature

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature
  Scenario: first scenario
    When a step is passing

  Scenario: second scenario
    When a step does not exist
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.When(/^a step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f progress features/a.feature:2` 👍 (296ms)

Then

it outputs this text: 👍 (000ms)

```
.

1 scenario (1 passed)
1 step (1 passed)
```

And

the exit status should be 0 👍 (000ms)

Scenario: run a single feature without step definitions

Given

a file named "features/a.feature" with: 👍 (001ms)

```
Feature: some feature
  Scenario:
    When a step is undefined
```

When

I run `cucumber.js -f progress features/a.feature` 👍 (189ms)

Then

it outputs this text: 👍 (000ms)

```
U

1 scenario (1 undefined)
1 step (1 undefined)

You can implement step definitions for undefined steps with these snippets:

this.When(/^a step is undefined$/, function (callback) {
  // Write code here that turns the phrase above into concrete actions
  callback.pending();
});
```

And

the exit status should be 0 👍 (000ms)

Scenario: run feature with non-default step definitions file location specified (-r option)

Given

a file named "features/a.feature" with: 👍 (002ms)

```
Feature: some feature
  Scenario:
    When a step is passing
```

And

a file named "step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.When(/^a step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f progress features/a.feature -r step_definitions/cucumber_steps.js` 👍 (196ms)

Then

it outputs this text: 👍 (000ms)

```
.

1 scenario (1 passed)
1 step (1 passed)
```

And

the exit status should be 0 👍 (000ms)

Scenario: run feature with step definitions in required directory (-r option)

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature
  Scenario:
    When a step is passing
```

And

a file named "step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.When(/^a step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f progress features/a.feature -r step_definitions` 👍 (285ms)

Then

it outputs this text: 👍 (000ms)

```
.

1 scenario (1 passed)
1 step (1 passed)
```

And

the exit status should be 0 👍 (000ms)

Scenario: display Cucumber version

When

I run `cucumber.js --version` 👍 (102ms)

Then

I see the version of Cucumber 👍 (000ms)

And

the exit status should be 0 👍 (000ms)

Scenario: display help

When

I run `cucumber.js --help` 👍 (097ms)

Then

I see the help of Cucumber 👍 (000ms)

And

the exit status should be 0 👍 (000ms)

Scenario: display help (short flag)

When

I run `cucumber.js -h` 👍 (196ms)

Then

I see the help of Cucumber 👍 (000ms)

And

the exit status should be 0 👍 (000ms)

Scenario: run a single failing feature

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature
  Scenario:
    When a step is failing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.When(/^a step is failing$/, function(callback) { callback("forced error");
});
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f progress features/a.feature` 👍 (296ms)

Then

it outputs this text: 👍 (000ms)

```
F

(::) failed steps (::)

forced error

Failing scenarios:
<current-directory>/features/a.feature:2 # Scenario:

1 scenario (1 failed)
1 step (1 failed)
```

And

the exit status should be 1 👍 (000ms)

Scenario: run a single failing feature with an empty hooks file

Given

a file named "features/a.feature" with: 👍 (002ms)

```
Feature: some feature
  Scenario:
    When a step is failing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.When(/^a step is failing$/, function(callback) { callback("forced error");
});
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

When

I run `cucumber.js -f progress features/a.feature` 👍 (199ms)

Then

it outputs this text: 👍 (000ms)

```
F

(::) failed steps (::)

forced error

Failing scenarios:
<current-directory>/features/a.feature:2 # Scenario:

1 scenario (1 failed)
1 step (1 failed)
```

And

the exit status should be 1 👍 (000ms)

Scenario: run a single failing feature with an AfterFeatures hook

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: some feature
  Scenario:
    When a step is failing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.When(/^a step is failing$/, function(callback) { callback("forced error");
});
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function() {
  this.registerHandler('AfterFeatures', function (event, callback) {
    callback();
  });
};
module.exports = hooks;
```

When

I run `cucumber.js -f progress features/a.feature` 👍 (269ms)

Then

it outputs this text: 👍 (000ms)

```
F

(::) failed steps (::)

forced error

Failing scenarios:
<current-directory>/features/a.feature:2 # Scenario:

1 scenario (1 failed)
1 step (1 failed)
```

And

the exit status should be 1 👍 (000ms)

Core feature elements execution

In order to have automated acceptance tests
As a developer
I want Cucumber to run core feature elements

Scenario: Simple flat steps

Given

a step definition matching /^a step passes$/ 👍 (000ms)

When

I run the following feature: 👍 (005ms)

```
Feature: Simple flat steps
  In order to execute features
  As cucumber
  I want to run features successfully

  Scenario: Simple flat step
    Given a step passes
    When a step passes
    Then a step passes
```

Then

the feature should have run successfully 👍 (000ms)

Scenario: Given, When, Then, And and But steps

Given

a "Given" step definition matching /^a "Given" step passes$/ 👍 (000ms)

And

a "When" step definition matching /^a "When" step passes$/ 👍 (000ms)

And

a "Then" step definition matching /^a "Then" step passes$/ 👍 (000ms)

When

I run the following feature: 👍 (011ms)

```
Feature: Given, When, Then, And and But step execution
  Scenario: All kinds of steps
    Given a "Given" step passes
    When a "When" step passes
    Then a "Then" step passes

  Scenario: All kinds of steps with And's and But's
    Given a "Given" step passes
    And a "Given" step passes
    But a "Given" step passes
    When a "When" step passes
    And a "When" step passes
    But a "When" step passes
    Then a "Then" step passes
    And a "Then" step passes
    But a "Then" step passes
```

Then

the feature should have run successfully 👍 (000ms)

Scenario: Step definition body is executed

Given

a step definition matching /^I call a watched step$/ counting its calls 👍 (000ms)

And

a step definition matching /^the watched step should have been called (\d+) times?$/ checking the number of step calls 👍 (000ms)

When

I run the following feature: 👍 (017ms)

```
Feature: Step definition body execution
  Scenario: Step definition body is executed once
    When I call a watched step
    Then the watched step should have been called 1 time

  Scenario: Step definition body is executed several times
    When I call a watched step
    And I call a watched step
    And I call a watched step
    Then the watched step should have been called 3 times
```

Then

the feature should have run successfully 👍 (000ms)

Scenario: Steps accepting parameters

Given

a step definition matching /^I call a step with "(.*)"$/ recording its parameters 👍 (000ms)

And

a step definition matching /^I call a step with "(**.)", "(.**)" and "(.*)"$/ recording its parameters 👍 (000ms)

And

a step definition matching /^the (\d+)(?:st|nd|rd) received parameter should be "(.*)"$/ checking a recorded parameter 👍 (000ms)

When

I run the following feature: 👍 (006ms)

```
Feature: Steps receiving parameters
  Scenario: Single-parameter step
    When I call a step with "a parameter"
    Then the 1st received parameter should be "a parameter"

  Scenario: Three-parameter step
    When I call a step with "one", "two" and "three"
    Then the 1st received parameter should be "one"
    And the 2nd received parameter should be "two"
    And the 3rd received parameter should be "three"
```

Then

the feature should have run successfully 👍 (000ms)

Scenario: Steps accepting a DocString parameter

Given

a step definition matching /^I call a step with the following text:$/ recording its parameters 👍 (000ms)

And

a step definition matching /^I call a step with "(.*)" and the following text:$/ recording its parameters 👍 (000ms)

And

a step definition matching /^the (\d+)(?:st|nd) received parameter should be "(.*)"$/ checking a recorded parameter 👍 (000ms)

And

a step definition matching /^the (\d+)(?:nd) received parameter should be:$/ checking a recorded parameter 👍 (000ms)

When

I run the following feature: 👍 (010ms)

```
Feature: Steps receiving a DocString parameter
  Scenario: One-liner DocString parameter
    When I call a step with the following text:
      """
      The cucumber (Cucumis sativus) is a widely cultivated plant in the gourd
family Cucurbitaceae.
      """
    Then the 1st received parameter should be "The cucumber (Cucumis sativus) is a
widely cultivated plant in the gourd family Cucurbitaceae."

  Scenario: Matching group and one-liner DocString
    When I call a step with "Cucumber" and the following text:
      """
      The cucumber (Cucumis sativus) is a widely cultivated plant in the gourd
family Cucurbitaceae.
      """
    Then the 1st received parameter should be "Cucumber"
    And the 2nd received parameter should be "The cucumber (Cucumis sativus) is a
widely cultivated plant in the gourd family Cucurbitaceae."

  Scenario: Matching group and multiline DocString
    When I call a step with "Cucumber" and the following text:
      """
      cu·cum·ber |'kyoō,kəmbər|
      noun
        1. a long, green-skinned fruit with watery flesh, usually eaten raw in
salads or pickled.
        2. the climbing plant of the gourd family that yields this fruit, native to
the Chinese Himalayan region. It is widely cultivated but very rare in the wild. •
Cucumis sativus, family Cucurbitaceae.
      """
    Then the 1st received parameter should be "Cucumber"
    And the 2nd received parameter should be:
      """
      cu·cum·ber |'kyoō,kəmbər|
      noun
        1. a long, green-skinned fruit with watery flesh, usually eaten raw in
salads or pickled.
        2. the climbing plant of the gourd family that yields this fruit, native to
the Chinese Himalayan region. It is widely cultivated but very rare in the wild. •
Cucumis sativus, family Cucurbitaceae.
      """
```

Then

the feature should have run successfully 👍 (000ms)

Data Tables

Scenario: a data table can be read as a hash

Given

the following data table in a step: 👍 (000ms)

```
| Cucumber     | Cucumis sativus |
| Burr Gherkin | Cucumis anguria |
```

When

the data table is passed to a step mapping that converts it to a hash 👍 (006ms)

Then

the data table is converted to the following: 👍 (000ms)

```
{ "Cucumber":"Cucumis sativus", "Burr Gherkin": "Cucumis anguria" }
```

Environment Hooks

Scenario: Tagged around hook with untagged scenario

Given

an around hook tagged with "@foo" 👍 (000ms)

When

Cucumber executes a scenario with no tags 👍 (004ms)

Then

the hook is not fired 👍 (000ms)

Scenario: Hooks are steps

Given

a file named "features/a.feature" with: 👍 (002ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.Before(function(callback) {
    callback();
  });

  this.After(function(callback) {
    callback();
  });

  this.Around(function(runScenario) {
    runScenario(function(callback) {
      callback();
    });
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (301ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
```

```
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Before ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
```

```
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: Failing around hook (pre scenario) fails the scenario

Given

a file named "features/a.feature" with: 👍 (005ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.Around(function(runScenario) {
    runScenario('Fail', function(callback) { callback(); });
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (302ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "error_message": "<error-message>",
              "duration": "<duration>",
              "status": "failed"
            },
            "match": {}
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
```

```
            "result": {
              "status": "skipped"
            },
            "match": {}
          },
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: Failing around hook (pre scenario) fails the scenario

Given

a file named "features/a.feature" with: 👍 (005ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.Around(function(runScenario) {
    runScenario.fail();
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (300ms)

Then

it outputs this json: 👍 (000ms)


```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "error_message": "<error-message>",
              "duration": "<duration>",
              "status": "failed"
            },
            "match": {}
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
```

```
            "result": {
              "status": "skipped"
            },
            "match": {}
          },
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: Failing around hook (post scenario) fails the scenario

Given

a file named "features/a.feature" with: 👍 (005ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.Around(function(runScenario) {
    // no-op

    runScenario(function(callback) {
      callback('Fail');
    });
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (240ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
```

```
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "error_message": "<error-message>",
              "duration": "<duration>",
              "status": "failed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: Failing around hook (post scenario) fails the scenario

Given

a file named "features/a.feature" with: 👍 (005ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.Around(function(runScenario) {
    // no-op

    runScenario(function(callback) {
      callback.fail();
    });
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (249ms)

Then

it outputs this json: 👍 (003ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
```

```
            "match": {}
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "error_message": "<error-message>",
              "duration": "<duration>",
              "status": "failed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: Failing before hook fails the scenario

Given

a file named "features/a.feature" with: 👍 (005ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.Before(function(callback) {
    callback('Fail');
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (236ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Before ",
            "hidden": true,
            "result": {
              "error_message": "<error-message>",
              "duration": "<duration>",
              "status": "failed"
            },
            "match": {}
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "status": "skipped"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```


Scenario: Failing before hook fails the scenario

Given

a file named "features/a.feature" with: 👍 (005ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.Before(function(callback) {
    callback.fail();
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (267ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Before ",
            "hidden": true,
            "result": {
              "error_message": "<error-message>",
              "duration": "<duration>",
              "status": "failed"
            },
            "match": {}
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "status": "skipped"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```


Scenario: Failing after hook fails the scenario

Given

a file named "features/a.feature" with: 👍 (005ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.After(function(callback) {
    callback('Fail');
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (200ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "error_message": "<error-message>",
              "duration": "<duration>",
              "status": "failed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: Failing after hook fails the scenario

Given

a file named "features/a.feature" with: 👍 (007ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.After(function(callback) {
    callback.fail();
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (304ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "error_message": "<error-message>",
              "duration": "<duration>",
              "status": "failed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: Hooks still execute after a failure

Given

a file named "features/a.feature" with: 👍 (005ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.Around(function(scenario, runScenario) {
    runScenario("fail", function(callback) {
      callback();
    });
  });

  this.Around(function(scenario, runScenario) {
    runScenario(function(callback) {
      callback();
    });
  });

  this.Before(function(scenario, callback) {
    callback();
  });

  this.After(function(scenario, callback) {
    callback();
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (207ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
```

```
        "type": "scenario",
        "steps": [
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "error_message": "<error-message>",
              "duration": "<duration>",
              "status": "failed"
            },
            "match": {}
          },
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Before ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "status": "skipped"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
```

```
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: World is this in hooks

Given

a file named "features/a.feature" with: 👍 (005ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/world.js" with: 👍 (001ms)

```
var WorldConstructor = function WorldConstructor(callback) {
  var world = {
    isWorld: function() { return true; }
  };

  callback(world); // tell Cucumber we're finished and to use our world object
instead of 'this'
};

module.exports.World = WorldConstructor;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.World = require("../support/world.js").World;

  this.Before(function(callback) {
    var world = this;

    if (!world.isWorld())
      callback("Expected this to be world");
    else
      callback();
  });

  this.After(function(callback) {
    var world = this;

    if (!world.isWorld())
      callback("Expected this to be world");
    else
      callback();
  });

  this.Around(function(runScenario) {
    var world = this;
    var error;

    if (!world.isWorld())
      error = "Expected this to be world";
    else
      error = null;

    runScenario(error, function(callback) {
      var world = this;
      var error;

      if (!world.isWorld())
        error = "Expected this to be world";
      else
        error = null;

      callback(error);
    });
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (200ms)

Then

it outputs this json: 👍 (000ms)


```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Before ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
```

```
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

JSON Formatter

In order to simplify processing of Cucumber features and results
Developers should be able to consume features as JSON

Scenario: output JSON for a feature with no scenarios

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature
```

When

I run `cucumber.js -f json` 👍 (282ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature"
  }
]
```

Scenario: output JSON for a feature with one undefined scenario

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

Scenario: I havn't done anything yet
```

When

I run `cucumber.js -f json` 👍 (294ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I havn't done anything yet",
        "id": "some-feature;i-havn't-done-anything-yet",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario"
      }
    ]
  }
]
```

Scenario: output JSON for a feature with one scenario with one undefined step

Given

a file named "features/a.feature" with: 👍 (001ms)

```
Feature: some feature

Scenario: I've declared one step but not yet defined it
    Given I have not defined this step
```

When

I run `cucumber.js -f json` 👍 (202ms)

Then

it outputs this json: 👍 (000ms)


```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri":"<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step but not yet defined it",
        "id": "some-feature;i've-declared-one-step-but-not-yet-defined-it",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name":"I have not defined this step",
            "line":4,
            "keyword":"Given ",
            "result":
            {
              "status":"undefined"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: output JSON for a feature with one undefined step and subsequent defined steps which should be skipped

Given

a file named "features/a.feature" with: 👍 (001ms)

```
Feature: some feature

Scenario: One pending step and two following steps which will be skipped
    Given This step is undefined
    Then this step should be skipped
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Then(/^this step should be skipped$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (307ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "One pending step and two following steps which will be skipped",
        "id": "some-feature;one-pending-step-and-two-following-steps-which-will-be-
skipped",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is undefined",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "status": "undefined"
            },
            "match": {
            }
          },
          {
            "name": "this step should be skipped",
            "line": 5,
            "keyword": "Then ",
            "result": {
              "status": "skipped"
            },
            "match": {
            }
          }
        ]
      }
    ]
  }
]
```

Scenario: output JSON for a feature with one scenario with one pending step

Given

a file named "features/a.feature" with: 👍 (002ms)

```
Feature: some feature

Scenario: I've declared one step which is pending
    Given This step is pending
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is pending$/, function(callback) { callback.pending(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (218ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri":"<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step which is pending",
        "id": "some-feature;i've-declared-one-step-which-is-pending",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is pending",
            "line": 4,
            "keyword": "Given ",
            "result": { "status": "pending" },
            "match": {
            }
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for a feature with one scenario with failing step

tags: @wip

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

Scenario: I've declared one step but it is failing
    Given This step is failing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is failing$/, function(callback) { callback.fail(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (214ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri":"<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step but it is failing",
        "id": "some-feature;i've-declared-one-step-but-it-is-failing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is failing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "error_message": "<error-message>",
              "duration": "<duration>",
              "status": "failed"
            },
            "match": {
            }
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for a feature with one scenario with passing step

Given

a file named "features/a.feature" with: 👍 (012ms)

```
Feature: some feature

Scenario: I've declared one step which passes
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (301ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri":"<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step which passes",
        "id": "some-feature;i've-declared-one-step-which-passes",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {
            }
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for a scenario with a passing step followed by one that is pending and one that fails

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: some feature

Scenario: I've declared one step which is passing, one pending and one failing.
    Given This step is passing
    And This step is pending
    And This step fails but will be skipped
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
  this.Given(/^This step is pending$/, function(callback) { callback.pending(); });
  this.Given(/^This step fails but will be skipped$/, function(callback) {
callback.fail(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (298ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step which is passing, one pending and one
failing.",
        "id": "some-feature;i've-declared-one-step-which-is-passing,-one-pending-
and-one-failing.",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
```

```
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "name": "This step is pending",
            "line": 5,
            "keyword": "And ",
            "result": {
              "status": "pending"
            },
            "match": {}
          },
          {
            "name": "This step fails but will be skipped",
            "line": 6,
            "keyword": "And ",
            "result": {
              "status": "skipped"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: output JSON for a scenario with a pending step followed by one that passes and one that fails

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: some feature

Scenario: I've declared one step which is passing, one pending and one failing.
    Given This step is pending
    And This step is passing but will be skipped
    And This step fails but will be skipped
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is pending$/, function(callback) { callback.pending(); });
  this.Given(/^This step is passing but will be skipped$/, function(callback) {
callback(); });
  this.Given(/^This step fails but will be skipped$/, function(callback) {
callback.fail(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (287ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step which is passing, one pending and one
failing.",
        "id": "some-feature;i've-declared-one-step-which-is-passing,-one-pending-
and-one-failing.",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
```

```
          {
            "name": "This step is pending",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "status": "pending"
            },
            "match": {
            }
          },
          {
            "name": "This step is passing but will be skipped",
            "line": 5,
            "keyword": "And ",
            "result": {
              "status": "skipped"
            },
            "match": {
            }
          },
          {
            "name": "This step fails but will be skipped",
            "line": 6,
            "keyword": "And ",
            "result": {
              "status": "skipped"
            },
            "match": {
            }
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for one feature, one passing scenario, one failing scenario

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: one passes one fails

Scenario: This one passes
  Given This step is passing
Scenario: This one fails
  Given This step is failing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
  this.Given(/^This step is failing$/, function(callback) { callback.fail(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (321ms)

Then

it outputs this json: 👍 (000ms)


```
[
  {
    "id": "one-passes-one-fails",
    "name": "one passes one fails",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "This one passes",
        "id": "one-passes-one-fails;this-one-passes",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
```

```
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {
          }
        }
      ]
    },
    {
      "name": "This one fails",
      "id": "one-passes-one-fails;this-one-fails",
      "line": 5,
      "keyword": "Scenario",
      "description": "",
      "type": "scenario",
      "steps": [
        {
          "name": "This step is failing",
          "line": 6,
          "keyword": "Given ",
          "result": {
            "error_message": "<error-message>",
            "duration": "<duration>",
            "status": "failed"
          },
          "match": {
          }
        }
      ]
    }
  ]
}
]
```

Scenario: output JSON for multiple features

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: feature a

Scenario: This is the first feature
    Given This step is passing
```

And

a file named "features/b.feature" with: 👍 (001ms)

```
Feature: feature b

Scenario: This is the second feature
    Given This step is passing
```

And

a file named "features/c.feature" with: 👍 (001ms)

```
Feature: feature c

Scenario: This is the third feature
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json features/a.feature features/b.feature features/c.feature` 👍 (290ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "feature-a",
    "name": "feature a",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "This is the first feature",
```

```
      "id": "feature-a;this-is-the-first-feature",
      "line": 3,
      "keyword": "Scenario",
      "description": "",
      "type": "scenario",
      "steps": [
        {
          "name": "This step is passing",
          "line": 4,
          "keyword": "Given ",
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {
          }
        }
      ]
    }
  ]
},
{
  "id": "feature-b",
  "name": "feature b",
  "description": "",
  "line": 1,
  "keyword": "Feature",
  "uri": "<current-directory>/features/b.feature",
  "elements": [
    {
      "name": "This is the second feature",
      "id": "feature-b;this-is-the-second-feature",
      "line": 3,
      "keyword": "Scenario",
      "description": "",
      "type": "scenario",
      "steps": [
        {
          "name": "This step is passing",
          "line": 4,
          "keyword": "Given ",
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {}
        }
      ]
```

```
      }
    ]
  },
  {
    "id": "feature-c",
    "name": "feature c",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/c.feature",
    "elements": [
      {
        "name": "This is the third feature",
        "id": "feature-c;this-is-the-third-feature",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for multiple features each with multiple scenarios

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: feature a

Scenario: This is the feature a scenario one
    Given This step is passing

Scenario: This is the feature a scenario two
    Given This step is passing

Scenario: This is the feature a scenario three
    Given This step is passing
```

And

a file named "features/b.feature" with: 👍 (001ms)

```
Feature: feature b

Scenario: This is the feature b scenario one
    Given This step is passing

Scenario: This is the feature b scenario two
    Given This step is passing

Scenario: This is the feature b scenario three
    Given This step is passing
```

And

a file named "features/c.feature" with: 👍 (001ms)

```
Feature: feature c

Scenario: This is the feature c scenario one
    Given This step is passing

Scenario: This is the feature c scenario two
    Given This step is passing

Scenario: This is the feature c scenario three
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json features/a.feature features/b.feature features/c.feature` 👍 (320ms)

Then

it outputs this json: 👍 (003ms)

```
[
  {
    "id": "feature-a",
    "name": "feature a",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "This is the feature a scenario one",
        "id": "feature-a;this-is-the-feature-a-scenario-one",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {
            }
          }
        ]
      },
      {
        "name": "This is the feature a scenario two",
        "id": "feature-a;this-is-the-feature-a-scenario-two",
```

```
      "line": 6,
      "keyword": "Scenario",
      "description": "",
      "type": "scenario",
      "steps": [
        {
          "name": "This step is passing",
          "line": 7,
          "keyword": "Given ",
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {}
        }
      ]
    },
    {
      "name": "This is the feature a scenario three",
      "id": "feature-a;this-is-the-feature-a-scenario-three",
      "line": 9,
      "keyword": "Scenario",
      "description": "",
      "type": "scenario",
      "steps": [
        {
          "name": "This step is passing",
          "line": 10,
          "keyword": "Given ",
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {}
        }
      ]
    }
  ]
},
{
  "id": "feature-b",
  "name": "feature b",
  "description": "",
  "line": 1,
  "keyword": "Feature",
  "uri": "<current-directory>/features/b.feature",
  "elements": [
    {
```

```
      "name": "This is the feature b scenario one",
      "id": "feature-b;this-is-the-feature-b-scenario-one",
      "line": 3,
      "keyword": "Scenario",
      "description": "",
      "type": "scenario",
      "steps": [
        {
          "name": "This step is passing",
          "line": 4,
          "keyword": "Given ",
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {}
        }
      ]
    },
    {
      "name": "This is the feature b scenario two",
      "id": "feature-b;this-is-the-feature-b-scenario-two",
      "line": 6,
      "keyword": "Scenario",
      "description": "",
      "type": "scenario",
      "steps": [
        {
          "name": "This step is passing",
          "line": 7,
          "keyword": "Given ",
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {}
        }
      ]
    },
    {
      "name": "This is the feature b scenario three",
      "id": "feature-b;this-is-the-feature-b-scenario-three",
      "line": 9,
      "keyword": "Scenario",
      "description": "",
      "type": "scenario",
      "steps": [
        {
```

```
          "name": "This step is passing",
          "line": 10,
          "keyword": "Given ",
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {}
        }
      ]
    }
  ]
},
{
  "id": "feature-c",
  "name": "feature c",
  "description": "",
  "line": 1,
  "keyword": "Feature",
  "uri": "<current-directory>/features/c.feature",
  "elements": [
    {
      "name": "This is the feature c scenario one",
      "id": "feature-c;this-is-the-feature-c-scenario-one",
      "line": 3,
      "keyword": "Scenario",
      "description": "",
      "type": "scenario",
      "steps": [
        {
          "name": "This step is passing",
          "line": 4,
          "keyword": "Given ",
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {}
        }
      ]
    },
    {
      "name": "This is the feature c scenario two",
      "id": "feature-c;this-is-the-feature-c-scenario-two",
      "line": 6,
      "keyword": "Scenario",
      "description": "",
      "type": "scenario",
```

```
        "steps": [
          {
            "name": "This step is passing",
            "line": 7,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      },
      {
        "name": "This is the feature c scenario three",
        "id": "feature-c;this-is-the-feature-c-scenario-three",
        "line": 9,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is passing",
            "line": 10,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for a feature with a background

Given

a file named "features/a.feature" with: 👍 (005ms)

```
Feature: some feature

Background:
  Given This applies to all scenarios
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This applies to all scenarios$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (294ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "",
        "keyword": "Background",
        "description": "",
        "type": "background",
        "line": 3,
        "steps": [
          {
            "name": "This applies to all scenarios",
            "line": 4,
            "keyword": "Given "
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for a feature with a failing background

Since the background step is re-evaluated for each scenario that is where the result of the step is currently recorded in the JSON output.

If the background is being re-evaluated for each scenario then it would be misleading to only output the result for the first time it was evaluated.

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

Background:
    Given This applies to all scenarios but fails
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This applies to all scenarios but fails$/, function(callback) {
callback.fail(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (289ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "",
        "keyword": "Background",
        "description": "",
        "type": "background",
        "line": 3,
        "steps": [
          {
            "name": "This applies to all scenarios but fails",
            "line": 4,
            "keyword": "Given "
          }
        ]
      }
    ]
  }
]
```

Scenario: output JSON for a feature with a DocString

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

Scenario: Scenario with DocString
  Given we have this DocString:
  """
  This is a DocString
  """
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^we have this DocString:$/, function(string, callback) { callback();
});
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (273ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "Scenario with DocString",
        "id": "some-feature;scenario-with-docstring",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "we have this DocString:",
            "line": 4,
            "keyword": "Given ",
            "doc_string":
              {
                "value": "This is a DocString",
                "line": 5,
                "content_type": ""
              },
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for background step with a DocString

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

Background: Background with DocString
  Given we have this DocString:
  """
  This is a DocString
  """
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^we have this DocString:$/, function(string, callback) { callback();
});
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (279ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "Background with DocString",
        "keyword": "Background",
        "description": "",
        "type": "background",
        "line": 3,
        "steps": [
          {
            "name": "we have this DocString:",
            "line": 4,
            "keyword": "Given ",
            "doc_string": {
              "value": "This is a DocString",
              "line": 5,
              "content_type": ""
            }
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for a feature with tags

Given

a file named "features/a.feature" with: 👍 (003ms)

```
@alpha @beta @gamma
Feature: some feature

Scenario: This scenario has no tags
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (303ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 2,
    "keyword": "Feature",
    "tags": [
      {
        "name": "@alpha",
        "line": 1
      },
      {
        "name": "@beta",
        "line": 1
      },
      {
        "name": "@gamma",
        "line": 1
      }
    ],
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "This scenario has no tags",
        "id": "some-feature;this-scenario-has-no-tags",
        "line": 4,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is passing",
            "line": 5,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: output JSON for scenario with tags

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

@one @two @three
Scenario: This scenario has tags
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (305ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "This scenario has tags",
        "id": "some-feature;this-scenario-has-tags",
        "line": 4,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "tags": [
          {
            "name": "@one",
            "line": 3
          },
          {
            "name": "@two",
            "line": 3
          },
          {
            "name": "@three",
            "line": 3
          }
        ],
        "steps": [
          {
            "name": "This step is passing",
            "line": 5,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: output JSON for a step with table

Rows do not appear to support line attribute yet.

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

Scenario: This scenario contains a step with a table
    Given This table:
    |col 1|col 2|col 3|
    |one  |two  |three|
    |1    |2    |3    |
    |!    |~    |@    |
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This table:$/, function(table, callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (262ms)

Then

it outputs this json: 👍 (002ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "This scenario contains a step with a table",
        "id": "some-feature;this-scenario-contains-a-step-with-a-table",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This table:",
            "line": 4,
            "keyword": "Given ",
            "rows": [
              { "cells": ["col 1", "col 2", "col 3" ] },
              { "cells": ["one", "two", "three"] },
              { "cells": ["1", "2", "3"] },
              { "cells": ["!", "~", "@"] }
            ],
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for background with table

Rows do not appear to support line attribute yet.

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

Background:
  Given This table:
    |col 1|col 2|col 3|
    |one  |two  |three|
    |1    |2    |3    |
    |!    |~    |@    |
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This table:$/, function(table, callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (202ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "",
        "keyword": "Background",
        "description": "",
        "type": "background",
        "line": 3,
        "steps": [
          {
            "name": "This table:",
            "line": 4,
            "keyword": "Given ",
            "rows": [
              { "cells": ["col 1", "col 2", "col 3"] },
              { "cells": ["one", "two", "three"] },
              { "cells": ["1", "2", "3"] },
              { "cells": ["!", "~", "@"] }
            ]
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for a feature with one scenario outline with no examples tables

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

  Scenario Outline: I've declared one step which passes
    Given This <instance> step is passing
```

When

I run `cucumber.js -f json` 👍 (301ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature"
  }
]
```

Scenario: output JSON for a feature with one scenario outline with an examples table with no rows

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

  Scenario Outline: I've declared one step which passes
    Given This <instance> step is passing

  Examples:
    | instance |
```

When

I run `cucumber.js -f json` 👍 (303ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature"
  }
]
```

Scenario: output JSON for a feature with one scenario outline with an examples table with two rows

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

  Scenario Outline: I've declared one step which passes
    Given This <instance> step is passing

  Examples:
    | instance |
    | first    |
    | second   |
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This (first|second) step is passing$/, function(instance, callback) {
callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (309ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step which passes",
        "id": "some-feature;i've-declared-one-step-which-passes",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This first step is passing",
```

```
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      },
      {
        "name": "I've declared one step which passes",
        "id": "some-feature;i've-declared-one-step-which-passes",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This second step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: output JSON for a feature with one scenario outline with an examples table with two rows and a background

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature
  Background:
    Given This applies to all scenarios

  Scenario Outline: I've declared one step which passes
    Given This <instance> step is passing

  Examples:
    | instance |
    | first    |
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This applies to all scenarios$/, function(callback) { callback(); });
  this.Given(/^This (first|second) step is passing$/, function(instance, callback) {
callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (247ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "",
        "keyword": "Background",
        "description": "",
        "type": "background",
        "line": 2,
        "steps": [
          {
```

```
            "name": "This applies to all scenarios",
            "line": 3,
            "keyword": "Given "
          }
        ]
      },
      {
        "name": "I've declared one step which passes",
        "id": "some-feature;i've-declared-one-step-which-passes",
        "line": 5,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This applies to all scenarios",
            "line": 3,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "name": "This first step is passing",
            "line": 6,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for a feature with one scenario outline with two examples tables

Given

a file named "features/a.feature" with: 👍 (002ms)

```
Feature: some feature

  Scenario Outline: I've declared one step which passes
    Given This <instance> step is passing

  Examples:
    | instance |
    | first    |

  Examples:
    | instance |
    | second   |
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This (first|second) step is passing$/, function(instance, callback) {
callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f json` 👍 (306ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step which passes",
        "id": "some-feature;i've-declared-one-step-which-passes",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
```

```
        "steps": [
          {
            "name": "This first step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      },
      {
        "name": "I've declared one step which passes",
        "id": "some-feature;i've-declared-one-step-which-passes",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This second step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```


Scenario: output JSON for a feature with one scenario outline with an examples table with two rows and before, after and around hooks

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

  Scenario Outline: I've declared one step which passes
    Given This <instance> step is passing

  Examples:
    | instance |
    | first    |
    | second   |
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This (first|second) step is passing$/, function(instance, callback) {
callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.Before(function(callback) {
    callback();
  });

  this.After(function(callback) {
    callback();
  });

  this.Around(function(runScenario) {
    runScenario(function(callback) {
      callback();
    });
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (268ms)

Then

it outputs this json: 👍 (002ms)


```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step which passes",
        "id": "some-feature;i've-declared-one-step-which-passes",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Before ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "name": "This first step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
```

```
        },
        {
          "keyword": "After ",
          "hidden": true,
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {}
        },
        {
          "keyword": "Around ",
          "hidden": true,
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {}
        }
      ]
    },
    {
      "name": "I've declared one step which passes",
      "id": "some-feature;i've-declared-one-step-which-passes",
      "line": 3,
      "keyword": "Scenario",
      "description": "",
      "type": "scenario",
      "steps": [
        {
          "keyword": "Around ",
          "hidden": true,
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {}
        },
        {
          "keyword": "Before ",
          "hidden": true,
          "result": {
            "duration": "<duration>",
            "status": "passed"
          },
          "match": {}
        },
        {
```

```
            "name": "This second step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

PogoScript support

In order to use the JS dialect that totally rocks
As a step definition implementor
I want to use PogoScript for writing step definitions

Scenario: PogoScript step definition

Given

a mapping written in PogoScript 👎 (000ms)

When

Cucumber executes a scenario using that mapping 👎 (000ms)

Then

the feature passes 👎 (000ms)

And

the mapping is run 👎 (000ms)

Pretty Formatter

In order to visualize the tests in an a set of Cucumber features
Developers should be able to see prettified view of the scenarios that are being executed

Scenario: Output pretty text for a feature with no scenarios

Given

a file named "features/a.feature" with: 👍 (002ms)

```
Feature: some feature
```

When

I run `cucumber.js -f pretty` 👍 (299ms)

Then

it outputs this text: 👍 (000ms)

```
Feature: some feature



0 scenarios
0 steps
```

Scenario: Pretty formatter hides around, before and after hooks

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

Scenario: I've declared one step which passes
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.Before(function(callback) {
    callback();
  });

  this.After(function(callback) {
    callback();
  });

  this.Around(function(runScenario) {
    runScenario(function(callback) {
      callback();
    });
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f pretty` 👍 (306ms)

Then

it outputs this text: 👍 (000ms)

```
Feature: some feature



  Scenario: I've declared one step which passes   # features/a.feature:3
    Given This step is passing                    # features/a.feature:4


1 scenario (1 passed)
1 step (1 passed)
```

Scenario: Failing hook is reported as a failed step

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.Before(function(callback) {
    callback('Fail');
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f pretty` 👍 (196ms)

Then

it outputs this text: 👍 (000ms)

```
Feature: some feature



  Scenario: I've declared one step and it is passing   # features/a.feature:3
    Before
      Fail
    Given This step is passing                         # features/a.feature:4


(::) failed steps (::)

Fail

Failing scenarios:
<current-directory>/features/a.feature:3 # Scenario: I've declared one step and it
is passing

1 scenario (1 failed)
2 steps (1 failed, 1 skipped)
```

Scenario Outlines and Examples

Scenario: Basic outline

Given

the following feature: 👍 (000ms)

```
Feature: testing scenarios
  Background:
    Given a background step

  Scenario Outline: A <some> step is followed by <result> steps
    When a <some> step
    Then i get <result>
  Examples:
    | some    | result  |
    | passing | passed  |
    | failing | skipped |
```

And

the step "a background step" has a passing mapping 👍 (000ms)

And

the step "a passing step" has a passing mapping 👍 (000ms)

And

the step "a failing step" has a failing mapping 👍 (000ms)

And

the step "i get passed" has a passing mapping 👍 (000ms)

And

the step "i get skipped" has a passing mapping 👍 (000ms)

When

Cucumber runs the feature 👍 (027ms)

Then

the scenario called "A failing step is followed by skipped steps" is reported as failing 👍 (000ms)

And

the step "a background step" passes 👍 (000ms)

And

the step "a passing step" passes 👍 (000ms)

And

the step "a failing step" passes 👍 (000ms)

And

the step "i get passed" passes 👍 (000ms)

And

the step "i get skipped" is skipped 👍 (000ms)

Scenario: Outline with table

Given

the following feature: 👍 (000ms)

```
Feature: testing scenarios
  Scenario Outline: outline with table
    When a table step:
      | first   | second   |
      | <first> | <second> |
  Examples:
    | first   | second  |
    | 1       | 2       |
```

And

the step "a table step:" has a passing mapping that receives a data table 👍 (000ms)

When

Cucumber runs the feature 👍 (006ms)

Then

the received data table array equals the following: 👍 (000ms)

```
[["first","second"],["1","2"]]
```

Scenario: Outline with doc string

Given

the following feature: 👍 (000ms)

```
Feature: testing scenarios
  Scenario Outline: outline with doc string
    When a doc string step:
      """
      I am doc string in <example> example
      And there are <string> string
      """
  Examples:
    | example | string |
    | first   | some   |
```

And

the step "a doc string step:" has a passing mapping that receives a doc string 👍 (000ms)

When

Cucumber runs the feature 👍 (005ms)

Then

the received doc string equals the following: 👍 (000ms)

```
I am doc string in first example
And there are some string
```

Scenario: Several outlines

Given

the following feature: 👍 (000ms)

```
Feature: testing scenarios
  Scenario Outline: scenario outline 1
    When step <id>

  Examples:
    | id |
    | a  |
    | b  |

  Scenario Outline: scenario outline 2
    When step <id>

  Examples:
    | id |
    | c  |
    | d  |
```

And

the step "step a" has a passing mapping 👍 (000ms)

And

the step "step b" has a passing mapping 👍 (000ms)

And

the step "step c" has a passing mapping 👍 (000ms)

And

the step "step d" has a passing mapping 👍 (000ms)

When

Cucumber runs the feature 👍 (009ms)

Then

the step "step a" passes 👍 (000ms)

And

the step "step b" passes 👍 (000ms)

And

the step "step c" passes 👍 (000ms)

And

the step "step d" passes 👍 (000ms)

Scenario Statuses

Scenario: Check scenario statuses

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
function checkScenarioStatuses(scenario) {
  var error;

  if (scenario.isSuccessful() !== true)
    error = "Expected isSuccessful to be true";
  else if (scenario.isFailed() !== false)
    error = "Expected isFailed to be false";
  else if (scenario.isPending() !== false)
    error = "Expected isPending to be false";
  else if (scenario.isUndefined() !== false)
    error = "Expected isUndefined to be false";
  else if (scenario.getException() !== null)
    error = "Expected exception to be null";
  else
    error = null;

  return error;
}

var hooks = function () {
  this.Around(function(scenario, runScenario) {
    var error = checkScenarioStatuses(scenario);

    runScenario(error, function(callback) {
      var error = checkScenarioStatuses(scenario);

      callback(error);
    });
  });

  this.Before(function(scenario, callback) {
    var error = checkScenarioStatuses(scenario);

    callback(error);
  });

  this.After(function(scenario, callback) {
    var error = checkScenarioStatuses(scenario);

    callback(error);
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (295ms)

Then

it outputs this json: 👍 (000ms)


```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Before ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
```

```
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "Around ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: Success status

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: some feature

Scenario: I've declared one step and it is passing
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.After(function(scenario, callback) {
    if (scenario.isSuccessful() !== true)
      error = "Expected isSuccessful to be true";
    else if (scenario.isFailed() !== false)
      error = "Expected isFailed to be false";
    else if (scenario.isPending() !== false)
      error = "Expected isPending to be false";
    else if (scenario.isUndefined() !== false)
      error = "Expected isUndefined to be false";
    else if (scenario.getException() !== null)
      error = "Expected exception to be null";
    else
      error = null;

    callback(error);
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (289ms)

Then

it outputs this json: 👍 (000ms)

```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is passing",
        "id": "some-feature;i've-declared-one-step-and-it-is-passing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is passing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```


Scenario: Failed status

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature

Scenario: I've declared one step and it is failing
    Given This step is failing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is failing$/, function(callback) { callback("Fail"); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.After(function(scenario, callback) {
    if (scenario.isSuccessful() !== false)
      error = "Expected isSuccessful to be false";
    else if (scenario.isFailed() !== true)
      error = "Expected isFailed to be true";
    else if (scenario.isPending() !== false)
      error = "Expected isPending to be false";
    else if (scenario.isUndefined() !== false)
      error = "Expected isUndefined to be false";
    else if (scenario.getException() !== "Fail")
      error = "Expected exception to be 'Fail'";
    else
      error = null;

    callback(error);
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (189ms)

Then

it outputs this json: 👍 (000ms)


```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is failing",
        "id": "some-feature;i've-declared-one-step-and-it-is-failing",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is failing",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "error_message": "<error_message>",
              "duration": "<duration>",
              "status": "failed"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: Pending status

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: some feature

Scenario: I've declared one step and it is pending
    Given This step is pending
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is pending$/, function(callback) { callback.pending(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.After(function(scenario, callback) {
    if (scenario.isSuccessful() !== false)
      error = "Expected isSuccessful to be false";
    else if (scenario.isFailed() !== false)
      error = "Expected isFailed to be false";
    else if (scenario.isPending() !== true)
      error = "Expected isPending to be true";
    else if (scenario.isUndefined() !== false)
      error = "Expected isUndefined to be false";
    else if (scenario.getException() !== null)
      error = "Expected exception to be null";
    else
      error = null;

    callback(error);
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (295ms)

Then

it outputs this json: 👍 (001ms)


```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is pending",
        "id": "some-feature;i've-declared-one-step-and-it-is-pending",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is pending",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "status": "pending"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: Undefined status

Given

a file named "features/a.feature" with: 👍 (005ms)

```
Feature: some feature

Scenario: I've declared one step and it is undefined
    Given This step is undefined
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.After(function(scenario, callback) {
    if (scenario.isSuccessful() !== false)
      error = "Expected isSuccessful to be false";
    else if (scenario.isFailed() !== false)
      error = "Expected isFailed to be false";
    else if (scenario.isPending() !== false)
      error = "Expected isPending to be false";
    else if (scenario.isUndefined() !== true)
      error = "Expected isUndefined to be true";
    else if (scenario.getException() !== null)
      error = "Expected exception to be null";
    else
      error = null;

    callback(error);
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (296ms)

Then

it outputs this json: 👍 (000ms)


```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is undefined",
        "id": "some-feature;i've-declared-one-step-and-it-is-undefined",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
        "steps": [
          {
            "name": "This step is undefined",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "status": "undefined"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "duration": "<duration>",
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Scenario: Simultaneous statuses

Given

a file named "features/a.feature" with: 👍 (004ms)

```
Feature: some feature

Scenario: I've declared one step and it is undefined
    Given This step is pending
    And This step is undefined
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is pending$/, function(callback) { callback.pending(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (000ms)

```
var hooks = function () {
  this.After(function(scenario, callback) {
    if (scenario.isSuccessful() !== false)
      error = "Expected isSuccessful to be false";
    else if (scenario.isFailed() !== true)
      error = "Expected isFailed to be true";
    else if (scenario.isPending() !== true)
      error = "Expected isPending to be true";
    else if (scenario.isUndefined() !== true)
      error = "Expected isUndefined to be true";
    else
      error = null;

    callback(error);
  });

  this.After(function(scenario, callback) {
    callback("fail");
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f json` 👍 (299ms)

Then

it outputs this json: 👍 (000ms)


```
[
  {
    "id": "some-feature",
    "name": "some feature",
    "description": "",
    "line": 1,
    "keyword": "Feature",
    "uri": "<current-directory>/features/a.feature",
    "elements": [
      {
        "name": "I've declared one step and it is undefined",
        "id": "some-feature;i've-declared-one-step-and-it-is-undefined",
        "line": 3,
        "keyword": "Scenario",
        "description": "",
        "type": "scenario",
```

```
        "steps": [
          {
            "name": "This step is pending",
            "line": 4,
            "keyword": "Given ",
            "result": {
              "status": "pending"
            },
            "match": {}
          },
          {
            "name": "This step is undefined",
            "line": 5,
            "keyword": "And ",
            "result": {
              "status": "undefined"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "error_message": "fail",
              "duration": 351161,
              "status": "failed"
            },
            "match": {}
          },
          {
            "keyword": "After ",
            "hidden": true,
            "result": {
              "duration": 319244,
              "status": "passed"
            },
            "match": {}
          }
        ]
      }
    ]
  }
]
```

Step definition callbacks

Scenario: fail with single-parameter error (Node.js style)

Given
a scenario with: 👍 (000ms)

```
When I divide 10 by 0
```

And
the step "I divide 10 by 0" has a mapping failing via a Node-like error construct 👍 (000ms)

When
Cucumber executes the scenario 👍 (009ms)

Then
the scenario fails 👍 (000ms)

Scenario: succeed with promise

Given
a promise-based mapping 👎 (000ms)

When
Cucumber executes that mapping 👎 (000ms)

Then
the mapping is run 👎 (000ms)

And
the scenario passes 👎 (000ms)

Scenario: fail with promise

Given
a failing promise-based mapping 👎 (000ms)

When
Cucumber executes that mapping 👎 (000ms)

Then
the mapping is run 👎 (000ms)

And
the scenario fails 👎 (000ms)

Scenario: succeed synchronously

Given
a passing synchronous mapping 👎 (000ms)

When
Cucumber executes that mapping 👎 (000ms)

Then
the mapping is run 👎 (000ms)

And
the scenario passes 👎 (000ms)

step definition snippets

Scenario: escape regexp special characters

Given

a scenario with: 👍 (000ms)

```
Given I am a happy veggie \o/
When I type -[]{}()*+?.\^$|#/
```

When

Cucumber executes the scenario 👍 (006ms)

Then

a "Given" step definition snippet for /^I am a happy veggie \\o\/$/ is suggested 👎 (000ms)

Then

a "When" step definition snippet for /^I type \-\[\]\{\}\(\)*\+\?\.\\\^\$\|#\/$/ is suggested 👎 [small right]#(000ms)#

Scenario: step matching groups

Given

a scenario with: 👍 (000ms)

```
Given I have 5 "kekiri" cucumbers
```

When

Cucumber executes the scenario 👍 (004ms)

Then

a "Given" step definition snippet for /^I have (\d+) "([^"]*)" cucumbers$/ with 2 parameters is suggested 👎 (000ms)

Scenario: multiple matching groups

Given

a scenario with: 👍 (000ms)

```
Given I have some "hekiri", "wild" and "regular" cucumbers
```

When

Cucumber executes the scenario 👍 (004ms)

Then

a "Given" step definition snippet for /^I have some "(<strong class="^"">)", "([^"])" and "([^"]*)" cucumbers$/ with 3 parameters is suggested   (000ms)

Scenario: outline steps with examples

Given

a scenario with: 👍 (000ms)

```
Given I have <some> cucumbers
```

When

Cucumber executes the scenario 👍 (005ms)

Then

a "Given" example step definition snippet for /^I have "(.*)" cucumbers$/ with 1 parameters is suggested 👍 (000ms)

step definitions with string pattern

Some people don’t like Regexps as step definition patterns.
Cucumber also supports string-based patterns.

Scenario: step definition with string-based pattern

Given

a mapping with a string-based pattern 👍 (000ms)

When

Cucumber executes a scenario using that mapping 👍 (004ms)

Then

the feature passes 👍 (000ms)

And

the mapping is run 👍 (000ms)

Scenario: step definition with string-based pattern and parameters

Given

a mapping with a string-based pattern and parameters 👍 (000ms)

When

Cucumber executes a scenario that passes arguments to that mapping 👍 (003ms)

Then

the feature passes 👍 (000ms)

And

the mapping is run 👍 (000ms)

And

the mapping receives the arguments 👍 (000ms)

Scenario: step definition with string-based pattern and multiple parameters

Given

a mapping with a string-based pattern and multiple parameters 👍 (000ms)

When

Cucumber executes a scenario that passes multiple arguments to that mapping 👍 (003ms)

Then

the feature passes 👍 (000ms)

And

the mapping is run 👍 (000ms)

And

the mapping receives the multipple arguments 👎 (000ms)

Strict mode

Using the `--strict` flag will cause cucumber to fail unless all the
step definitions have been defined.

Scenario: Succeed scenario with implemented step with --strict

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: Missing
  Scenario: Missing
    Given this step passes
```

Given

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (000ms)

```
var cucumberSteps = function() {
  this.When(/^this step passes$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f progress features/a.feature --strict` 👍 (292ms)

Then

it outputs this text: 👍 (000ms)

```
.

1 scenario (1 passed)
1 step (1 passed)
```

And

the exit status should be 0 👍 (000ms)

Scenario: Fail scenario with undefined step with --strict

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: Missing
  Scenario: Missing
    Given this step passes
```

When

I run `cucumber.js -f progress features/a.feature --strict` 👍 (292ms)

Then

it outputs this text: 👍 (000ms)

```
U

1 scenario (1 undefined)
1 step (1 undefined)

You can implement step definitions for undefined steps with these snippets:

this.Given(/^this step passes$/, function (callback) {
  // Write code here that turns the phrase above into concrete actions
  callback.pending();
});
```

And

the exit status should be 1 👍 (000ms)

Scenario: Fail Scenario with pending step with --strict

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: Missing
  Scenario: Missing
    Given this step passes
```

Given

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^this step passes$/, function(callback) { callback.pending(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f progress features/a.feature --strict` 👍 (214ms)

Then

it outputs this text: 👍 (000ms)

```
P

1 scenario (1 pending)
1 step (1 pending)
```

And

the exit status should be 1 👍 (000ms)

Scenario: Fail scenario with undefined step with -S

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: Missing
  Scenario: Missing
    Given this step passes
```

When

I run `cucumber.js -f progress features/a.feature -S` 👍 (244ms)

Then

it outputs this text: 👍 (000ms)

```
U

1 scenario (1 undefined)
1 step (1 undefined)

You can implement step definitions for undefined steps with these snippets:

this.Given(/^this step passes$/, function (callback) {
  // Write code here that turns the phrase above into concrete actions
  callback.pending();
});
```

And

the exit status should be 1 👍 (000ms)

Scenario: Fail Scenario with pending step with -S

Given

a file named "features/a.feature" with: 👍 (002ms)

```
Feature: Missing
  Scenario: Missing
    Given this step passes
```

Given

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^this step passes$/, function(callback) { callback.pending(); });
};
module.exports = cucumberSteps;
```

When

I run `cucumber.js -f progress features/a.feature -S` 👍 (301ms)

Then

it outputs this text: 👍 (000ms)

```
P

1 scenario (1 pending)
1 step (1 pending)
```

And

the exit status should be 1 👍 (000ms)

Summary Formatter

In order to get a quick overview of Cucumber test run
Developers should be able to see a high level summary of the scenarios that were executed

Scenario: Output summary for a feature with no scenarios

Given

a file named "features/a.feature" with: 👍 (003ms)

```
Feature: some feature
```

When

I run `cucumber.js -f summary` 👍 (213ms)

Then

it outputs this text: 👍 (000ms)

```
0 scenarios
0 steps
```

Scenario: Summary formatter hides around, before and after hooks

Given

a file named "features/a.feature" with: 👍 (002ms)

```
Feature: some feature

Scenario: I've declared one step which passes
    Given This step is passing
```

And

a file named "features/step_definitions/cucumber_steps.js" with: 👍 (001ms)

```
var cucumberSteps = function() {
  this.Given(/^This step is passing$/, function(callback) { callback(); });
};
module.exports = cucumberSteps;
```

And

a file named "features/support/hooks.js" with: 👍 (001ms)

```
var hooks = function () {
  this.Before(function(callback) {
    callback();
  });

  this.After(function(callback) {
    callback();
  });

  this.Around(function(runScenario) {
    runScenario(function(callback) {
      callback();
    });
  });
};

module.exports = hooks;
```

When

I run `cucumber.js -f summary` 👍 (291ms)

Then

it outputs this text: 👍 (000ms)

```
1 scenario (1 passed)
1 step (1 passed)
```

User logs into the system

In order to be able to use eraNET components
As a user
I want to log in to the system

Scenario: Unlogged user sees welcome page with login

Given

I have not logged or have logged out before 👎 (000ms)

When

I visit initial page 👎 (000ms)

Then

Default app should be loaded 👎 (000ms)

And

I should see login request 👎 (000ms)

And

I should not see any username 👎 (000ms)

Scenario: Minimal user sees welcome page with its username and logout

Given

I have logged as guest named "Guest" 👎 (000ms)

When

I visit initial page 👎 (000ms)

Then

Default app should be loaded 👎 (000ms)

And

I should see logout request 👎 (000ms)

And

I should see "Guest" as username 👎 (000ms)

Scenario: Unlogged user logs in

Given

I have not logged or have logged out before 👎 (000ms)

And

I have visited initial page 👎 (000ms)

And

I have seen login request 👎 (000ms)

When

I ask to log in 👎 (000ms)

Then

I should be taken to login page 👎 (000ms)

Scenario: Logged user logs out

Given

I have logged as guest named "Guest" 👎 (000ms)

And

I have visited initial page 👎 (000ms)

And

I have seen logout request 👎 (000ms)

When

I ask to log out 👎 (000ms)

Then

I should be taken to logout page 👎 (000ms)

Scenario: Disconnected user sees welcome page and reconnect option

Given

I have had a broken connection with api site 👎 (000ms)

When

I visit initial page 👎 (000ms)

Then

Default app should be loaded 👎 (000ms)

And

I should see reconnect request 👎 (000ms)

And

I should not see any username 👎 (000ms)

Scenario: User sees 'connecting' while connecting

Given

I have had lagging api site 👎 (000ms)

When

I visit initial page 👎 (000ms)

Then

I should see connecting message 👎 (000ms)

And

I should not see any username 👎 (000ms)

World constructor callback with object

It is possible for the World constructor function to tell Cucumber
to use another object than itself as the World instance:

```
this.World = function WorldConstructor(callback) {
var myCustomWorld = { dance: function() { /* ... */ } };
callback(myCustomWorld); // tell Cucumber to use myCustomWorld
// as the world object.
};
```

If no parameter is passed to the callback, the WorldConstructor
instance will be used by Cucumber:

```
this.World = function WorldConstructor(callback) {
var myCustomWorld = {};
callback(); // could have been written callback(this);
};
```

Scenario: scenario calling function on explicit world instance

Given

a custom World constructor calling back with an explicit object 👍 (000ms)

When

Cucumber executes a scenario that calls a function on the explicit World object 👍 (005ms)

Then

the feature passes 👍 (000ms)

And

the explicit World object function should have been called 👍 (000ms)