Table of Contents

[Goal 1](#_Toc453416934)

[Design 1](#_Toc453416935)

[Thought 1](#_Toc453416936)

[Plan 1](#_Toc453416937)

[Tasks 1](#_Toc453416938)

[Project structure 2](#_Toc453416939)

[Vision 2](#_Toc453416940)

[Environment 3](#_Toc453416941)

[Tool stack 3](#_Toc453416942)

[Dependencies 3](#_Toc453416943)

[Modules 4](#_Toc453416944)

[Program procedure 5](#_Toc453416945)

[module: Core 5](#_Toc453416946)

[Project details 5](#_Toc453416947)

[Knowledge 6](#_Toc453416948)

[Style 6](#_Toc453416949)

[Tricks 6](#_Toc453416950)

[evaluation 6](#_Toc453416951)

[advantage 6](#_Toc453416952)

[disadvantage or risk 6](#_Toc453416953)

[scenes 6](#_Toc453416954)

## Goal

Understand the project jQuery, from whole to part.

## Design

### Thought

## Plan

### Tasks

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **module** | **task** | | **description** | **time** | **status** |
| **Project structure** | Vision | | Project vision | 1h | Done! |
| Environment | | Running environment | Done! |
| Tool stack | | Dev tool, test tool, deploy tool.  (a tool stack flow and a function table) | Done! |
| Dependencies | | Find all the dependents of the project, and know those dependencies’ functions.  (dependency chart and a function table) | 0h | Done! |
| Modules | | Distinct modules and find the relationship of all modules.  (dependency chart and a function table) | 2h | Done! |
| **Program procedure** | ${module} | Core data structure | The core data structures used in the project. |  |  |
| Core algorithm | The core data algorithms used in the project. |  |  |
| **Project details** | ${module} | Logic |  |  |  |
| Knowledge | The knowledge you do not know |  |  |
| Style | Style of project: comment, naming, lint, closure. Slips, law. |  |  |
| Tricks | Programing tricks. |  |  |
| Other |  |  |  |
| **evaluation** | Advantage | | The advantages of this project |  |  |
| Disadvantage | | The disadvantage of this project |
| Scenes | | The scenes it suits and not suits |

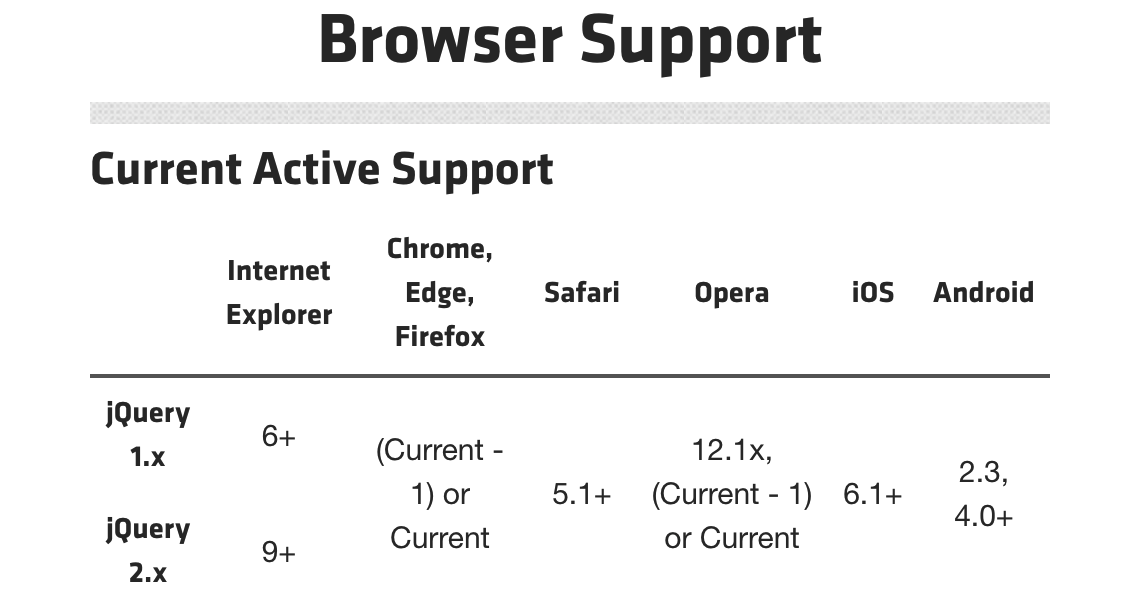
## Project structure

### Vision

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

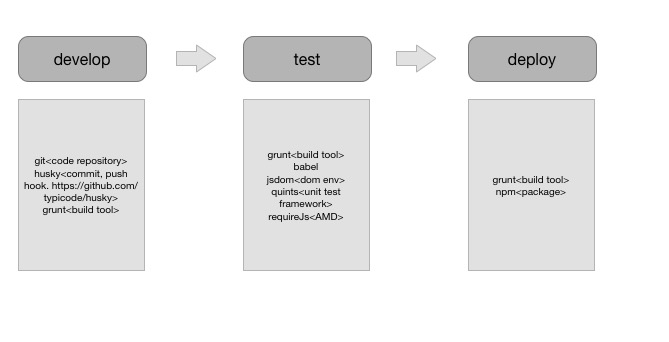
### Environment

* Browser Support



* jQuery also supports Node, browser extensions and other non-browser environments.

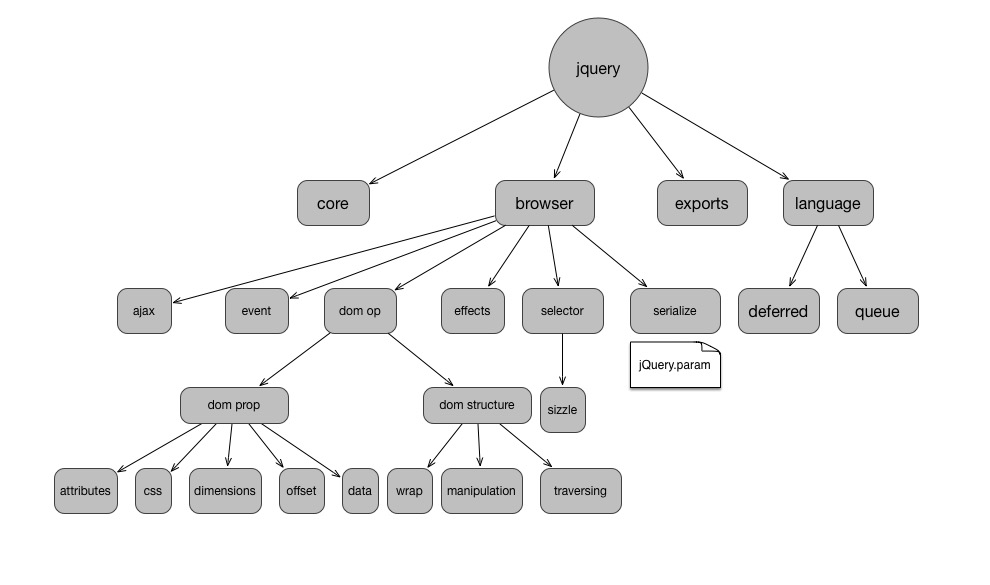
### Tool stack



### Dependencies

|  |  |
| --- | --- |
| **Module name** | **Function description** |
|  |  |

### Modules

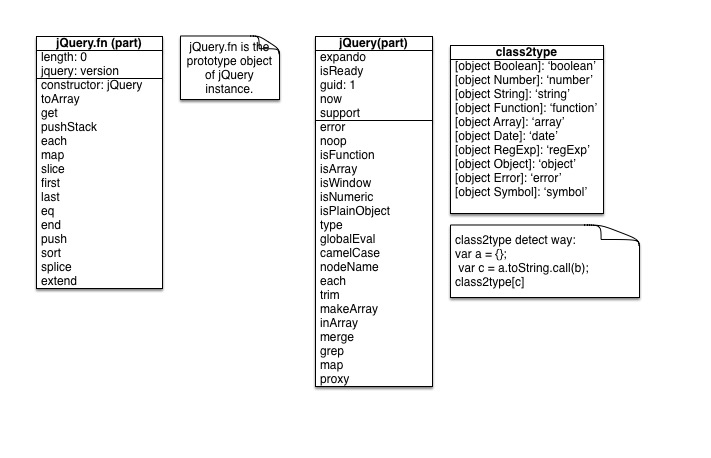


|  |  |
| --- | --- |
| **Module name** | **Function description** |
| **core** | jQuery constructor. jQuery.fn. Some array-like operations. jQuery.extend. Basic tool functions. |
| **ajax** | ajax request, ajax settings. |
| **event** | event model. on, off, dispatch, event object, etc. |
| **effects** | animation, fadeTo, stop, animate, finish |
| **selector** | Sizzle. Css selector engine. |
| **serialize** | jQuery.param. Serialize an array of form elements or a set of key/values into a query string. |
| **deferred** | promise |
| **attributes** | dom attr, prop, classes, val |
| **css** | css, style |
| **dimensions** | innerHeight, innerWidth, height, width, outerHeight, outerWidth |
| **offset** | offset, position, scrollLeft, scrollTop, top, left |
| **data** | data-\* attribute |
| **wrap** | wrapAll, wrapInner, wrap, unwrap |
| **manipulation** | Basic dom writing operation. text, remove, detach, append, prepend … |
| **traversing** | Some dom nearby reading operation. parent, parents, next, prev, sibling, children… |

## Program procedure

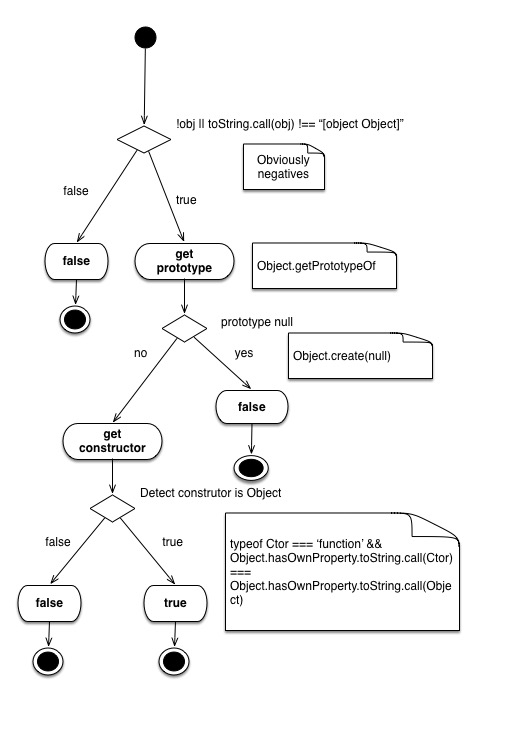
### module: Core

#### data structure



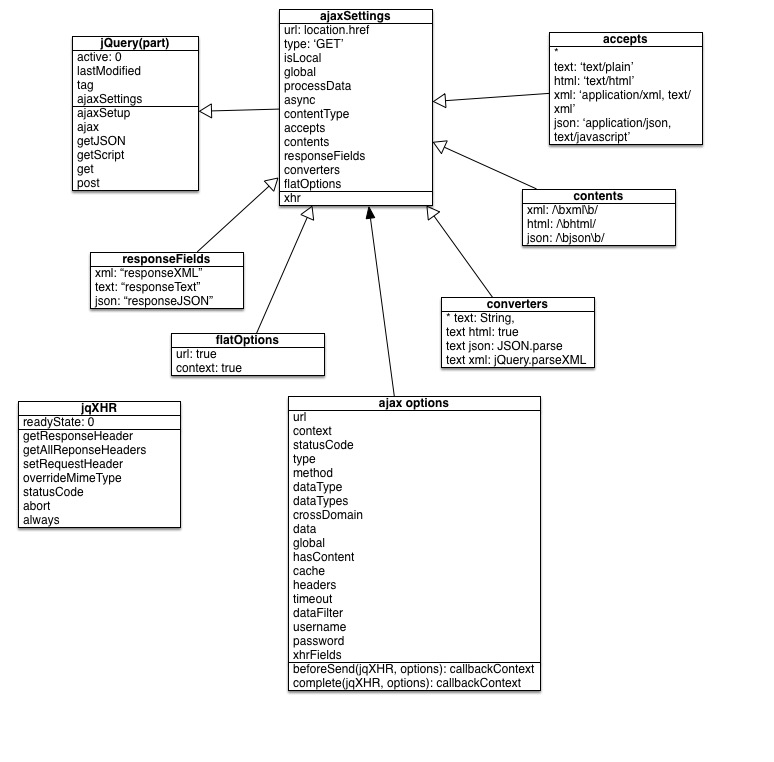
#### algorithm

* isPlainObject



### module: Ajax

#### data structure



* dataTypeExpression

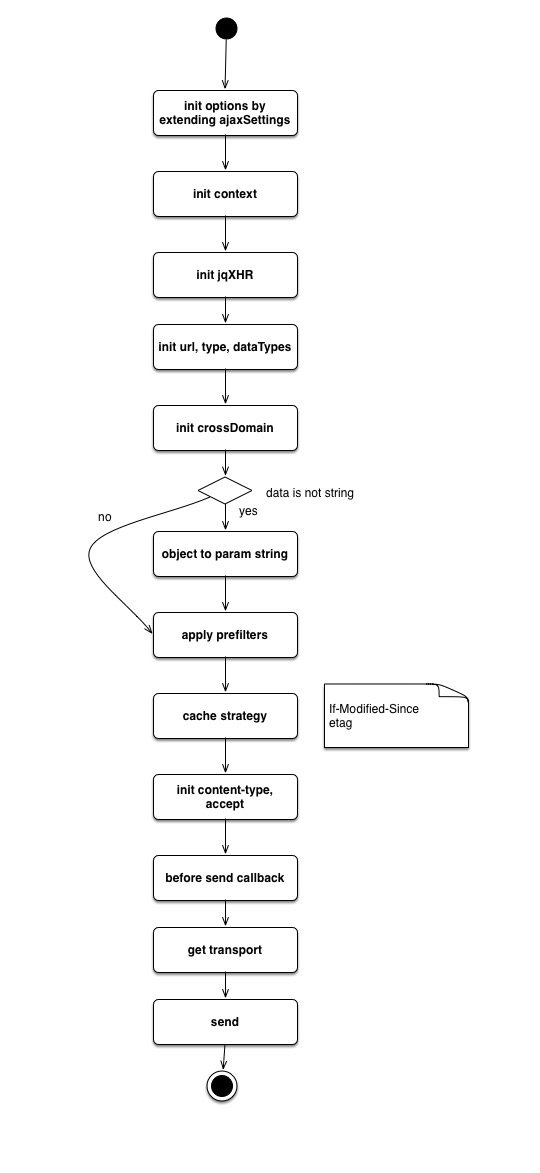
*datatypes* = *datatype* | *datatype* *whitespaces* *datatypes*

*datatype* ::= word | +word | \* | +

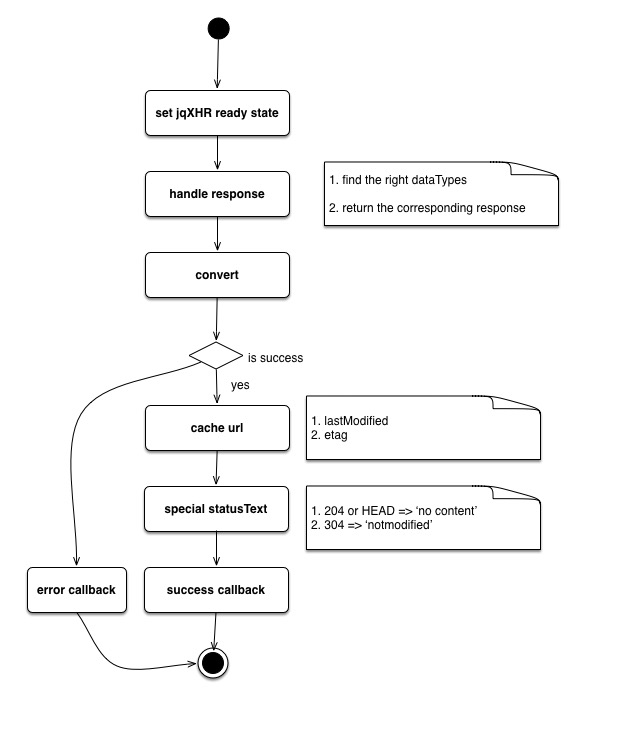
*whitespaces* ::= whitespace | whitespace *whitespaces*

#### algorithm

* core procedure
* ajax request process



* ajax response process



* inspect transports

1. execute every transport function by dataType
2. if result is string, inspect result as a dataType

## Project details

### Logic

|  |  |  |
| --- | --- | --- |
| **module** | **name** | **description** |
| **core** | **jQuery.fn.get(num)** | num is null, return slice.call(this); if num < 0, return this.[num+this.length];if num > 0, return this.[num] |
| **jQuery.extend=jQuery.fn.extend(deep, target, copy)** | * default deep is false. deep is optional. * if only one argument if not count deep, target is this * ignore null/undefined values |
| **ajax** | **Success status code** | status >=200 && status < 300 || status === 304 |
| **prefilters: {}** | * key is the datatype * the catchall symbol \* can be used * execution will start with transport datatype and THEN go to \* if needed |
| **transports: {}** | * key is the datatype * the catchall symbol \* can be used * execution will start with transport datatype and THEN go to \* if needed |

### Knowledge

|  |  |  |
| --- | --- | --- |
| **module** | **name** | **description** |
| **core** | **DOMEval** | dynamic add and remove script node. |
| **BOM** |  |
| **NBSP** |  |
| **rtrim** | /^[\s\uFEFE\xA0]+|[\s\uFEFE\xA0]+$/g |
| **Object.create(null)** | Prototype is null |
| **ajax** | **detect type** | var a = {}; a.toString.call(true) === ‘[object Boolean]’ |
|  |  |  |

### Style

### Tricks

|  |  |  |
| --- | --- | --- |
| **module** | **target** | **method** |
| **core** |  |  |

## evaluation

### advantage

### disadvantage or risk

### scenes