mootools Core

Full CheatSheet for Javascript Framework mootools 1.3
by mediavrog.net/blog/

Core

instanceOf(item m, type m)
typeOf(item m)
element, textnode, number, whitespace, function, date, arguments, array, object, string, boolean, regexp, class, collection, window, document,event, false

Object: Browser

ie, ie6, ie7, ie8,
firefox, firefox2, firefox3,
safari, safari3, safari4,
chrome, opera

Features

xpath, xhr, air, query, json

Request

Platform

mac, win, linux, ios, webos, android, other, anyName

Plugins

Flash

Class

new Class(o | contructor fn)
special properties:
Extends: o | class | a,
Implements: o,
initialize: fn (=constructor)
implement(o)

Class.Extras

Class: Chain

new Class({Implements: Chain})
callChain([args])
chain(fn [, fn [,...]])
clearChain()

Class: Events

new Class({Implements: Events})
addEvent(s, fn [, internal?])
addEvents(o, fn [, internal?])
fireEvent(s [, args, delay ms])
removeEvent(s, fn)
removeEvents([s])

Class: Options

new Class({Implements: Options})
setOptions([opt])

Type: Object

Static methods (O = Object)

* O.each(fn(v, k, o){} [, bn])
* O.every(fn(v, k, o){} [, bn])
* O.filter(fn(v, k, o){} [, bn])
* O.keyOf(o, value m)
* O.map(fn(v, k, o){} [, bn])
* O.some(fn(v, k, o){} [, bn])

* mostly synonymous to Array fn

O.append(origin o, ext o)
O.clone(o)
O.contains(o, value m)
O.getLength(o)
O.keys(o)
O.merge(o1, o2 [, o3, ...])
O.subset(o, keys a)
O.toQueryString(o)
O.values(o)

Type: String

String.from(m)
String.uniqueID()
camelCase()
capitalize()
clean()
contains(s [, separator s])
escapeRegExp()
hyphenate()
stripScripts(evaluate?)
substitute(o [, regex])
test(regex [, params])
toInt(),
toFloat()
trim()
rgbToHex(returnArray?)
hexToRgb(returnArray?)

Type: Function

Function.from(m)
Function.attempt(fn [, fn [, ...]])
attempt([args [, bn])
bind([bn [, args]])
delay([ms [, bn [, args]]])
extend(key s, value m)
implement(key s, value m)
pass([args [, bn])
periodical([ms [, bn [, args]]])

replace: bindWithEvent

myEl.addEvent("click",function(e){
myFunction.bind(bn [, e])
})

replace: run

myFn.apply(bn, arg)

Type: Array

Array.each(iterable, fn [, bind])
Array.clone(a)
Array.from(o)
* each(fn(el, i, a){} [, bn])
* every(fn(el, i, a){} [, bn])
* filter(fn(el, i, a){} [, bn])
* indexOf(el [, from n])
* map(fn(el, i, a){} [, bn])
* some(fn(el, i, a){} [, bn])

* only if not supported natively

append(a)
associate(a)
clean()
combine(a)
contains(el [, from n])
erase(el)
empty()
flatten()
getLast()
getRandom()
include(el)
invoke(method [, arg, arg, ...])
link(o)
pick()
rgbToHex(returnArray?)
hexToRgb(returnArray?)

Type: Number

Number.from(m)
Number.random(min n, max n)
limit(min n, max n)
round([n])
times(fn [, bn])
toInt(), toFloat()

Methods from 'Math'

abs, acos, asin, atan2, ceil, cos, exp, floor, log, max, min, pow, sin, sqrt, tan

Type: Event

new Event([e [, win]])
Properties
alt, client.x, client.y, code, control, key, meta, page.x, page.y, shift, relatedTarget, rightClick, target, wheel
'key' can be:
enter, up, down, left, right, tab, space, backspace, delete, esc
preventDefault()
stop(), stopPropagation()

Object: Event.Keys

Event.Keys.key = keyCode

Element

Type: Element

new Element(tag s | el | selector s [, opt])
each opt calls 'Element.set'
getElement(match)
getElements(match)
getElementById(s)
set(s, val | o)
get(s)
erase(s)
match(match)
contains(el)
inject(el [, where])
<el>myEl</el> (move myEl)
grab(el [, where])
<myEl>el</myEl> (move el)
adopt(el [, el a | el [,...]])
<myEl>el el</myEl> (move el's)
wraps(el [, where])
<myEl>el</myEl> (move myEl)
appendText(s)
empty() remove children
destroy() trash, free memory
dispose() remove from DOM
clone([cloneContents?, keepId?])
replaces(el)
hasClass(s)
addClass(s)
removeClass(s)
toggleClass(s)
getPrevious([match])
getAllPrevious([match])
getNext([match])
getAllNext([match])
getFirst([match])
getLast([match])
getParent([match])
getParents([match])
getSiblings([match])
getChildren([match])
toQueryString()
getSelected() (only on <select>)
getProperty(s)
getProperties(s [,s [, ...]])
setProperty(s, val)
setProperties({s: val, ...})
removeProperty(s)
removeProperties(s [,s [, ...]])
store(s, val)
retreive(s [, default m])
eliminate(s)

Type: Window

document.id(el | s | o)
Alias: $
(function($){
// $ is safe in closure (compat!)
})(document.id)
$$(selector s | el a | el[, el, ...])
any combination; commaseparated

Object: Element.Properties

html: htmlStr [, htmlStr [, ...]]
text: textString
prop: propValue
tag (only getter)

Type: IFrame

new IFrame([el [, opt]])

Type: Elements

new Elements(el a [, opt])
filter(sel s)

Element.Style

Type: Element

setStyle(s, val)
setStyles({s : val, ...})
getStyle(s)
getStyles(s [, s [, ...]])

Element.Event

Type: Element

addEvent(e, fn)
addEvents({e: fn})
removeEvent(e, fn)
removeEvents([e])
fireEvent(e [, args, delay])
cloneEvents(from el [, type s])

Object: Element.Events

Element.Events.key = o
o = { base: e, condition: fn,
onAdd: fn, onRemove: fn }

Custom Events

mouseenter
mouseleave
mousewheel

Element.Dimensions

Type: Element

getCoordinates()
getOffsetParent()
getPosition(relative el)
setPosition({x: posX, y: posY})
getScroll(), getScrollSize()
getSize()
scrollTo(x,y)

Class: Request

new Request([opt])
opt = {
url: s,
method: post | get,
data: s,
link: ignore | cancel | chain,
async: asyncRequest?,
encoding: s, (default: utf-8)
headers: {name: content},
evalScripts: eval<script>?,
evalResponse: evalAll?,
emulation: *put*_method?,
urlEncoded: formUrlEncoding?,
timeout: ms,
noCache: forceNoCache?,
user: basicAuthUser s,
password: basicAuthPasswd s,
isSuccess: fn,
onRequest(),
onLoadStart(event, xhr),
onProgress(event, xhr),
onComplete(),
onCancel(),
onSuccess(rText, rXml),
onFailure(xhr),
onException(hdName ,val),
onTimeout()
}
cancel()
getHeader(name s)
setHeader(name s, val s)
isRunning()
send([Request opt])

Object: Element.Properties

send [, Request opt]

Type: Element

send([url s]) (only on <form>)

Object: Cookie

Cookie.write(key s, value s [, opt])
opt = {
domain: s, path: s,
duration: n, secure: b?
}
Cookie.read(key s)
Cookie.dispose(key s [, opt])

WindowEvent: domready

domready
the all time favourite ;)

Class: Request.HTML

new Request.HTML([opt])
opt = { all opt from Request +
update: el,
append: el,
evalScripts: eval<script>?,
filter: fn,
onSuccess(rTree, rElems, rHTML, rJS)
}
get(opt | url s)
post(opt | queryString | el)

Object: Element.Properties

load [, opt]

Type: Element

load(url s) > Request.HTML.get

Class: Request.JSON

new Request.JSON([opt])
opt = { all opt from Request +
secure: checkSyntax?
onComplete(rJSON, rText)
}

Object: JSON

JSON.encode(o)
JSON.decode(s [, secure?])

Class: Swiff

new Swiff(swfPath s [, opt])
opt = {
id: s
width: n, height : n,
container: el,
params: swfParams,
properties: o,
vars: o,
events: o
}
swfParams = {
allowScriptAccess: s,
quality: high | medium | low,
swLiveConnect: remoteScripting?,
wMode: s
}
Swiff.remote(mySwiff o, fn [, arg, arg, ...])

Class: Slick (Selectors)

'Slick' engine FTW!
https://github.com/mootools/slick

o	~ Object	e	~ Event	bn	~ Element to bind 'this'
s	~ String	fn	~ Function	[ ]	~ optional
a	~ Array	el	~ Element	|	~ choice / or
n	~ Number	el a	~ Array of el	ms	~ Milliseconds
?	~ Boolean	m	~ mixed	match	~ CSS Selector

mootools FX/More

Full CheatSheet for Javascript Framework mootools 1.3
by mediavrog.net/blog/

Class: FX

new Fx(opt)
opt = {
fps: n (default: 50),
unit: false | px | em | %,
link: ignore | cancel | chain,
duration: ms | short | normal | long,
transition: Fx.Transitions,
onStart(fxInstance),
onComplete(fxInstance),
onCancel(fxInstance),
onChainComplete(fxInstance)
}
start(from n, [to n])
set(value m n)
cancel()
pause()
resume()

Class: Fx.Tween

new Fx.Tween(el, opt)
opt = { all opt from Fx +
property: cssProp s }
set(cssProp s, value m)
start([cssProp s,] [from,] to)

Object: Element.Properties

tween, [opt]

Type: Element

tween(cssProp s, from [, to])
fade([how])
how = in | out | show | hide | toggle
or number between 0 and 1
highlight([start, end])

Class: FX.Morph

new Fx.Morph(el, opt)
opt = { all opt from Fx }
set(match | {cssProp: to})
start(match | {cssProp: from, [to] })

Object: Element.Properties

morph, [opt]

Type: Element

morph(match | {cssProp: from, [to] })

Fx.Transitions

Class: Fx
adds possibility to use transition option as string e.g. 'bounce:out'

Object: Fx.Transitions

Linear, Quad, Cubic, Quart, Quint, Pow, Expo, Circ, Sine, Back, Bounce, Elastic
each has easIn,easeOut,easeInOut

Class: Fx.Transition

new Fx.Transition(trans [, opt])

mootools More

From here on you will find some selected plugins from More, I consider useful.
It's not a complete list!
Be sure to check out mootools.net/docs/more for latest up-to-date information.

Found typos?

maik@mediavrog.net

Class: Fx.Elements

new Fx.Elements(el a, opt)
opt = all opt from FX
set(to)
to = {
index of el: {cssProp: to}
}
start(obj)
obj = {
index of el: {cssProp: [from, to]}
}

Class: Fx.Slide

new Fx.Slide(el, opt)
opt = { all opt from Fx +
mode: horizontal | vertical,
wrapper: el,
hideOverflow: setHidden?,
resetHeight: autoResetHeight?
}
slideIn([mode])
slideOut([mode])
toggle([mode])
hide([mode])
show([mode])

Hash: Element.Properties

slide, [opt]

Type: Element

slide([how])
how = in | out | show | hide | toggle

Class: Fx.Scroll

new Fx.Scroll(el, opt)
opt = { all opt from Fx +
offset: {x: n, y: n},
overflown: a,
wheelStops: wheelStopsTrans?
}
set(x, y)
start(x, y)
toTop(), toBottom()
toLeft(), toRight(),
toElement(el)

Class: Drag

new Drag(el, opt)
opt = {
grid: pixels n,
handle: el,
invert: invertValuesOnDrag?,
limit: {x: n, y:n},
modifiers: {x: cssProp, y: cssProp}
snap: distance n,
style: setModifiersAsStyleProp?
unit: s (default: px),
preventDefault: b?, > Event
stopPropagation: b?, > Event
onBeforeStart(el),
onStart(el),
onSnap(el),
onDrag(el),
onComplete(el),
onCancel(el)
}
attach()
detach()
stop([event])

Type: Element

makeResizable([opt])
opt = all opt from Drag

Class: Drag.Move

new Drag.Move(el, opt)
opt = { all opt from Drag +
container: el,
droppables: el a,
precalculate: b?,
includeMargins: b?,
checkDroppables: b?,
onDrop(el, droppable, event),
onLeave(el, droppable),
onEnter(el, droppable)
}
stop()

Type: Element

makeDraggable([opt])
opt = all opt from Drag / Drag.Move

Module: Types

Array.Extras (Type: Array)

min()
max()
average()
shuffle()
sum()
unique()
reduce(fn [, firstCallVal m])
reduceRight(fn [, firstCallVal m])
fn(previousVal, currentVal, i, a)

String.Extras (Type: String)

pad(length, padString, dir)
dir = left | right | both
repeat(times n)
tidy() common special-chars to ascii
standardize() remove non-ascii
getTags([tagType, contents])
stripTags([tagType, contents])

Object.Extras (Type: Object)

O.getFromPath(o, path s)
path like 'key1.sub1.sub3'
O.cleanValues(o, fn(val))
O.erase(o, key)
O.run(o [, arg [, arg [, ...]]])

Number.Format (Type: Number)

format([opt])
opt = {
decimal: separator s,
group: separator s,
decimals: numOfDecimals n,
precision: significantNum n,
scientific: replace'e+4'?,
prefix: s, suffix: s
}
formatCurrency() > Locale
formatPercentage()

Hash: Asset

Asset.javascript(source s [, opt])
Asset.css(source s [, opt])
opt = { all opt from Element +
onLoad()
}
Asset.image(source s [, opt])
opt = { all opt from Element +
onLoad(), onError(), onAbort()
}
Asset.images(sources a [, opt])
opt = { all opt from Element +
onComplete(),
onProgress(counter, index),
onError(counter, index)
}
Note: Don't use Mootools events!

Type: Date

get(key)
set(key, val) / set({key: val})
key = Date, Day, FullYear / year, Hours / hr, Milliseconds / ms, Minutes / min, Month / mo, Seconds / sec, Time, UTCDate, UTCFullYear, UTCHours, UTCMilliseconds, UTCMinutes, UTCMonth, UTCSeconds
for 'get(key)' key may also be: TimezoneOffset, Week, Timezone, GMTOffset, Ordinal, DayOfYear, LastDayOfMonth, UTCDay, AMPM
clone()
increment(resolution, times n)
decrement(resolution, times n)
diff(date [, resolution])
resolution = year, month, week, day, hour, minute, second, ms
isLeapYear()
clearTime()
toISOString()
parse(date | s)

Static methods (D = Date)

D.defineFormat(name, format)
D.defineFormats({name, format})
D.parse(date | s)
D.defineParser(pattern s)
D.defineParsers(pattern a)
pattern = hybrid of format keys & regular expressions
%key - match key
? - optional
() - groups
e.g. "%d%o(%b(%Y)?)?(%X)?"
~ 14th
~ 31st October
~ 1 Jan 2000
~ 1 Jan 12:00am
D.define2DigitYearStart(year)
example:
D.parse('01/01/00');//Year 2000
D.parse('12/31/99');//Year 1999
D.define2DigitYearStart(2000);
D.parse('01/01/00');//Year 2000
D.parse('12/31/99');//Year 2099

Class: Date.Extras

Extra Date Parsers

Date.parse("today")
Date.parse("next monday") ...

Type: Date

timeDiff([date, joiner])
timeDiffInWords([date])

Info: Date.format

format(format)

keys: ("%key %key2%key3")

a short day ("Mon", "Tue")
A full day ("Monday")
b short month ("Jan", "Feb")
B full month ("January")
c full date to string ("Mon Dec 10 14:35:42 2007")
d date to two digits (01, 05, ...)
e date as one digit (1, 5, 12, ...)
H hour to two digits / 24h (00 - 24)
I hour as decimal / 12h (01 - 12)
j day of the year to three digits (001 - 366, is Jan 1st)
k hour / 24h as a digit (0 - 23) Single digits preceded by space
l hour / 12h as digit (1 to 12). Single digits preceded by space
L time in milliseconds to 3 digits
m numerical month to two digits (01 is Jan, 12 is Dec)
M minutes to two digits (01 - 59)
o ordinal of the day of the month in the current language ("st" for 1st, "nd" for 2nd, etc.)
p current language equivalent of either AM or PM
s Unix Epoch Time timestamp
S seconds to two digits (01 - 59)
U week to two digits (01- 52)
w numerical day of week one digit (0 is Sunday, 1 is Monday)
x date in the current language preferred format en-US: %m/%d/%Y (12/10/2007)
X time in the current language preferred format en-US: %I:%M%p (02:45PM)
y short year in two digits ("07")
Y full year in four digits ("2007")
z GMT offset ("-0800")
Z time zone ("GMT")
% returns % (%y%% = 07%)

shortcuts:

db
%Y-%m-%d %H:%M:%S
compact
%Y%m%dT%H%M%S
iso8601
%Y-%m-%dT%H:%M:%S%T
rfc822
%a, %d %b %Y %H:%M:%S %Z
short
%d %b %H:%M
long
%B %d, %Y %H:%M

Module: Request

Class: Request.Periodical

extends Request, Request.HTML & Request.JSON
opt = { all opt from Request +
initialDelay: ms,
delay: ms,
limit: ms
}
startTimer(m)
stopTimer()

Class: Request.Queue

new Request.Queue(opt)
opt = {
stopOnFailure: b?,
autoAdvance: b?,
concurrent: parrallelReq n
} + Events from Request like so: onComplete(name, inst, rText, rXml)
addRequest(name, request)
addRequests({name, request})
cancel(name)
clear([name])
getName(request)
getRunning()
hasNext([name])
removeRequest(name | request)
resume()
runAll()
runNext([name])

Type: URI

new URI([strUri, opt])
opt = {
base: baseHref s
}
toString()
set(part, value)
part = scheme, user, password, host, port, directory, file, query, fragment, data
get(part)
setData(o [, merge?, part])
getData([key, part])
clearData()
go()
toURI()

Type: String

toURI()

Module: URI.Relative

Type: URI

toAbsolute()
toRelative()