Global Variables

- `amount` *(number):*
  - the number of spawned elements present; starts at 0;
  - gets reduced by 20 when it reaches 100 every time;
- `dragging` *(boolean):*
  - determines whether any elements are being dragged;
  - if true, only images will be spawned while dragging;
- `consoleLogging` *(boolean):*
  - set to false by default;
  - when set to true in the console, a message about the type (`scatter.type`) of spawned element will be logged at its spawning;
- `totalNumOfImg` *(number)*: the total number of images to select from in the image folder;

Classes/Objects

Scatter/scatter

(pseudo-randomly scatters the 6 sections of texts onload)

- `composition` (property - *object of objects*):
  - containing 9 different objects (for the 9 corresponding versions), with each one containing an array of top positions and an array of left positions;
  - each array of positions contains 6 numeric values for the 6 sections of texts to inherit from;
- `version` (property - *number*):
  - the randomly generated version number;
- `onload()` (method - *run onload*):
  - matches the randomly generated `version` to the corresponding composition object and assigns its top and left positions accordingly;

Spawn/spawn

(the parent class that determines all behaviors surrounding spawning)

- `type` (property - *string*):
  - 'screenshot' or 'images';
- `time` (property - *number*):
  - it determines the amount of timeout (in ms) before the next value for itself is assigned; set to 2000 ms by default;
  - the callback function for when timeout finishes determines randomly (between `maxTime` and `minTime`) the next timeout;
- `maxTime` / `minTime` (property - *number*):
  - the range between which time is generated;
- `interval` (property - *setTimeout Id*):

    - the extracted variable for the interval and could be used with clearInterval()
- `id` (property - *string*):
    - a property to be inherited;
    - generates the appropriate HTML id for the spawned element when every time a derived object is created
- `position` (property - *array*):
    - a property to be inherited;
    - an array containing the return values(top and left positions) from the method `spawn.setPosition()`;
    - used to set the random top and left positions of the spawned elements;
- `getTime()` (method - *run onload*):
    - the first kicker of the function chain;
    - sets the random interval, and calls `spawn.run()` in its callback function;
- `run()` (method):
    - randomly decides if an 'image' or a 'screenshot' should be created, then pass the result to `spawn.create()` as arguments;
    - increases the `amount` and runs `spawn.clear()` if amount reaches 100; sets draggable after calling `spawn.create()`;
- `create(type)` (method):
    - parses `spawn.run()`'s randomly generated type and passes it to the `assign()` method of the appropriate derived object;
- `setPosition()` (method) :
    - sets up an object that contains 3 ranges of values for when `amount` is ≥ 1, ≥ 7, or ≥ 20;
    - matches `amount` to the appropriate top / left positions and returns the generated value to `spawn.position`, which will later be used to set the element's position in `spawn.place()`;
- `place(type, node)` (method):
    - the primary method for creating and appending the appropriate element;
    - first creates the content container (div), gives it appropriate properties (position, class, id), then appends it to the body;
    - then it creates the spawning element depending on the `type`, giving it appropriate attributes and appends it to the content container;
    - the `node` argument is the `canvas` DOM node (from html2canvas); it is used when `type` = 'screenshot';
- `clear()` (method):
    - executed when `amount` reaches 100; subtracts 20 from `amount`;
    - uses a for loop to get the first 20 spawned elements from the DOM then removes them;

Screenshot (extends Scatter)

(the derived class for when `type` == 'screenshots')

- `width` / `height` (property - *string*):
  - the randomly generated width and height that's used to assign to `canvas` in `spawn.place()`;
- `assign()` (method):
  - executes html2canvas() on #wrapper;
  - calls `spawn.place()` with `canvas` as the 'screenshot' `type`, and executes `canvas.toDataURL()`;

Images (extends Scatter)

(the derived class for when `type` == 'images')

- `imgNum` (property - *number*):
  - randomly generates an image number ≤ `totalNumOfImg`;
- `src` (property - *string*):
  - uses the `imgNum` to generate the corresponding `src` string for the appropriate file in the image folder
- `width` (property - *string*):
  - the randomly generated width that's used to assign to `canvas` in `spawn.place()`;
- `assign()` (method):
  - calls `spawn.place()` as the 'images' `type`

Citation/citation

(assigns all event listeners for all citations and animations)

- `spanList` (property - *array of DOM nodes*):
  - an array of DOM node for all in-line citations
- `citationList` (property - *array of DOM nodes*):
  - an array of DOM node for all side citations
- `allCitations` (property - *DOM node*):
  - the DOM node for the citation wrapper
- `intro` (property - *DOM node*):
  - the DOM node for the intro
- `spanSpliced` (property - *array of DOM nodes*):
  - `splice`s 15a/15b out of the `spanList` array, then stores the `splice`d portion as a new array
- `citationSpliced` (property - *array of DOM nodes*):
  - `splice`s 15 out of the `citationList` array, then stores the `splice`d portion as a new array
- `handlers` (property - *object of arrays*):
  - an object containing arrays that indicate the right DOM element style name, the `mouseover` behavior, and the `mouseout` behavior

- `evtHandling(attached, target, cssStyle, over, out, self)` (method):
  - `attached`: the DOM node for the event target;
  - `target`: the style change target for the event handler's callback function;
  - `cssStyle`: the DOM element style name;
  - `over` / `out`: the string value for the `mouseover` / `mouseout` style change
  - `self`: default to false; if set to true, `target` also includes the `attached` element itself in addition;
  - the `evtHandling()` method dynamically assigns all its arguments into an `addEventListner()` function;
  - detects if `target` is an array and parses the data accordingly;
- `biDirectional(span, citation, ...css)` (method):
  - creates a shorthand to execute `citaion.evtHandling()` for bi-directional and `self` == true event listeners;
- `initialize()` (method - run onload):
  - runs various for loops for the DOM elements in this class onload, executing either `citaion.evtHandling()` or `citaion.biDirectional()` on them;

Control/control

(manages dev keyboard events)

- `evt` (property - *object*):
  - contains the keyboard event `keys` for all dev functions;
- `terminate()` (method - *event handler*):
  - stops spawning (clears `spawn.interval`)
- `resurrect()` (method - *event handler*):
  - restarts spawning if it has been terminated (calls `spawn.getTime()`)
- `next()` (method - *event handler*):
  - immediately spawns an element with `spawn.run()`;
- `clear()` (method - *event handler*):
  - clears all spawned elements;
- `log()` (method - *console log*):
  - styles all console log messages for all `control` functions;

Standalone Functions

- `draggableSet(node)` (function - *onload*):
  - the jQuery UI draggable function;
  - set onload and every time an element is spawned;