Tree traversal

An application maintains a structured tree of objects (nodes) which have a parent child relationship.

Each node in this structure has

* a unique id (GUID) as well as a reference to its parent ID.
* a “children” property which is a collection of sub nodes, all of which have the same object signature as their parent

Create a class which has static methods that will

1) as optimally as possible flatten the entire tree structure into a single one-dimensional collection
2) given a unique id for a node, return its parent object
3) given a unique id for a node(a) and an instance of a new node(b), insert b into the “children” collection of node a user defined index

DOM styling

An API call within the application, returns a list of string values which denote a set of “tags” assigned to a parent “message” record. These tag values are a combination of user defined and system assigned values. The latter of these two types always contain a pre-fix which denote the type of tag.

These prefixes are

SYS:
PAR:
ACT:

User defined tags have no such prefix

In the UI, these tags are afforded unique styling properties depending on the prefix assigned.

Create a class/function which will receive this string value and return a DOM element that

1) has a unique css class assigned to it depending on the prefix
2) has text content which is equal to the value of the string the API returned, with the prefix removed - eg. “SYS:Foo” would have the text value “foo”

CSS layout

The design for an applications UI requires that the layout of an element changes for smaller devices.

The element in question is a tabular data list of messages (think email and chat messages) that have the following column

* Date received
* Subject
* From
* Attachments
* Type

For devices with large enough screens (960px +) the layout is shown with one item per row -

```
eg:__________________________________________________________________
  | Date                  Subject                  From   Attachments   Type   |
  |__________________________________________________________________|
```

On smaller screens however - the layout changes to a more column based layout -

```
eg:________________________________________
  | From                          Type          |
  | Subject                                     |
  | Data                          Attachments   |
  |________________________________________|
```

Using what ever DOM structure, and CSS typing technique you like, create an example of an element which with switch layout types at the defined break-point.