tree()

Fetches the current task dependency tree - in the rare case that it is needed. Generally, tree() won't be used by gulp consumers, but it is exposed so the CLI can show the dependency graph of the tasks defined in a gulpfile.

Usage

Example gulpfile:

```
const { series, parallel } = require('gulp');

function one(cb) {
  // body omitted
  cb();
}

function two(cb) {
  // body omitted
  cb();
}

function three(cb) {
  // body omitted
  cb();
}

const four = series(one, two);

const five = series(four,
  parallel(three, function(cb) {
    // Body omitted
    cb();
  })
);

module.exports = { one, two, three, four, five };
```

Output for tree():

```
{
  label: 'Tasks',
  nodes: [ 'one', 'two', 'three', 'four', 'five' ]
}
```

Output for tree({ deep: true }):

```
{
  label: "Tasks",
  nodes: [
    {
      label: "one",
      type: "task",
      nodes: []
    },
    {
      label: "two",
      type: "task",
      nodes: []
    },
    {
      label: "three",
      type: "task",
      nodes: []
    },
    {
      label: "four",
      type: "task",
      nodes: [
        {
          label: "<series>",
          type: "function",
          branch: true,
          nodes: [
            {
              label: "one",
              type: "function",
              nodes: []
            },
            {
              label: "two",
              type: "function",
              nodes: []
            }
          ]
        }
      ]
    },
```

```
{
  label: "five",
  type: "task",
  nodes: [
    {
      label: "<series>",
      type: "function",
      branch: true,
      nodes: [
        {
          label: "<series>",
          type: "function",
          branch: true,
          nodes: [
            {
              label: "one",
              type: "function",
              nodes: []
            },
            {
              label: "two",
              type: "function",
              nodes: []
            }
          ]
        },
        {
          label: "<parallel>",
          type: "function",
          branch: true,
          nodes: [
            {
              label: "three",
              type: "function",
              nodes: []
            },
            {
              label: "<anonymous>",
              type: "function",
              nodes: []
            }
          ]
        }
      ]
```

```
      }
    ]
  }
 ]
}
```

Signature

```
tree([options])
```

Parameters

parameter	type	note
options	object	Detailed in Options below.

Returns

An object detailing the tree of registered tasks - containing nested objects with 'label' and 'nodes' properties (which is archycompatible).
Each object may have a type property that can be used to determine if the node is a task or function.
Each object may have a branch property that, when true, indicates the node was created using series() or parallel().

Options

name	type	default	note
deep	boolean	false	If true, the entire tree will be returned. When false, only top level tasks will be returned.