کلاس node

این کلاس یک المنت در react را نمایندگی می کند و شبیه به نمایندگی کردن یک نود در DOM است.

**Properties**

* **id (string):** Unique identifier for the node.
* **tagName (string):** HTML tag name of the node (e.g., "div", "span").
* **openTag (string):** Opening HTML tag string for the node, including attributes and actions.
* **closeTag (string):** Closing HTML tag string for the node (empty for self-closing tags).
* **attribs (object):** Object containing attributes of the node (e.g., class, style).
* **children (array):** Array of child nodes.
* **inlineStyles (object):** Object containing inline styles for the node (e.g., color, font-size).
* **classNames (array):** Array of class names applied to the node.
* **actions (object):** Object containing event handlers for the node (e.g., onClick).
* **type (string):** Internal type of the node (e.g., "element", "text").
* **styleJsx (StyleJsx object, optional):** Reference to a StyleJsx object for managing CSS styles (implementation likely in a separate file).
* **property (object):** Internal property object for storing data used for conditional rendering and other behaviours.

**خصوصیت های مهم**

attribs: تمامی attributeهای یک المان اعم از استایل ها، کلاس های css و رویدادها (مثلا onClick) در این خصوصیت ذخیره می شوند. مثال:

{

classNames: [

"row"

],

inlineStyles: {

minHeight: "10px",

backgroundColor: "yellow"

},

id: "0-0row"

}

inlineStyle: همان طور که از نام آن پیداست استایل های inline که در open tag ها در بخش style نوشته می شوند، در این خصوصیت ذخیره می شوند.

classNames: نام کلاس های css در این خصوصیت ذخیره می شود.

actions: رویدادها در این بخش ذخیره می شوند. مثال:

{

onClick: "(e)=>{\n\t\t\te.stopPropagation()\n\t\t\t\n\t\t\tsetRLiveSelectedElement(\’0-0\’)}"

}

styleJsx: هر گره یا نود می تواند فرزندانی داشته باشد. یکی از این فرزندان می تواند styleJsx باشد که براذخیره استایل های یک کامپوننت معمولا به عنوان فرزند المنت اول آن کامپوننت تعریف می شود. در این خصوصیت یک شی از نوع کلاس styleJsx قرار می گیرد.

{

id: "styleJsx",

type: "styleJsx",

cssClasses: {

\_0: {

"background-color": "gray"

}

}

}

property: این خصوصیت مانند خصوصیت های قبلی برای ذخیره کردن داده های مربوط به یک المنت در react نیست. در این خصوصیت ویژگی های مورد نظر ما که در همین برنامه تعریف می شوند را ذخیره می کنیم.

**Methods**

* getId()**:** Returns the node's ID.
* getType()**:** Returns the node's type.
* getFields()**:** Returns an object containing all the node's properties.
* getLastChildID()**:** Returns id of last child or -1 if the node has no children.
* addNode(child, type, classNames)**:** Adds a child node to the current node.
  + child (object)**:** Object representing the child node.
  + type (string)**:** Type of the child node ("element" or "text").
  + classNames (string, optional)**:** Class names to be applied to the child node.
  + **Returns:** The newly created child Node object.

**Inheritance**

* LiveNode inherits from the Node class (likely defined elsewhere), which provides basic node properties and methods.

**Relationships**

* LiveNode maintains a static reference (allNodes) that stores all created LiveNode instances keyed by their IDs.
* This allows for efficient updates to child nodes throughout the tree.

**Usage**

This class is likely used to create a representation of a dynamic UI element or component. You can create LiveNode instances, add child nodes, set attributes, styles, and actions, and then use the toJsx method to convert the entire LiveNode tree into JSX code that can be rendered by a React or similar library.

LiveNode methods

* addInlineStyle(style)**:** Adds inline styles to the node.
  + style (object)**:** Object containing CSS styles to be applied inline.
* addClassNames(classNames)**:** Adds class names to the node's class list.
  + classNames (string, array)**:** A string or array of class names to be added.
* removeClassNames(classNames)**:** Removes class names from the node's class list.
  + classNames (string, array)**:** A string or array of class names to be removed.
* addAttrib()**:** (Documentation not provided, likely for adding attributes).
* updateProperty(property)**:** Updates the node's internal property based on the provided property object.
  + property (object)**:** Object containing a property name and value for conditional rendering.
* static toJsx()**:** Static method that converts the entire LiveNode tree starting from the root node (ID "0") to JSX code.
  + **Returns:** A string representing the JSX code for the LiveNode tree.