

Package: NGLVieweR (via r-universe)

October 24, 2024

Title Interactive 3D Visualization of Molecular Structures

Version 1.3.4

Maintainer Niels van der Velden <n.s.j.vandervelden@gmail.com>

Description Provides an 'htmlwidgets' <<https://www.htmlwidgets.org/>> interface to 'NGL.js' <<http://nglviewer.org/ngl/api/>>. 'NGLvieweR' can be used to visualize and interact with protein databank ('PDB') and structural files in R and Shiny applications. It includes a set of API functions to manipulate the viewer after creation in Shiny.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Imports htmlwidgets, magrittr, tools, shiny

Suggests knitr, webshot, markdown, rmarkdown

VignetteBuilder knitr

URL <https://github.com/nvelden/NGLVieweR>

BugReports <https://github.com/nvelden/NGLVieweR/issues>

Repository <https://nvelden.r-universe.dev>

RemoteUrl <https://github.com/nvelden/nglviewer>

RemoteRef HEAD

RemoteSha 811b4e159d6583756c58358f06f693a1ee50e98d

Contents

addRepresentation	2
addSelection	4
NGLVieweR	6
NGLVieweR-shiny	9
NGLVieweR_example	11

removeSelection	12
selectionParameters	13
setFocus	14
setPosition	15
setQuality	16
setRock	17
setRotation	17
setScale	18
setSpin	19
snapShot	20
stageParameters	21
updateColor	22
updateFocus	24
updateFullscreen	25
updateRepresentation	26
updateRock	28
updateSelection	29
updateSpin	31
updateStage	32
updateVisibility	34
updateZoomMove	35
zoomMove	37

Index	39
--------------	-----------

addRepresentation	<i>Add representation</i>
-------------------	---------------------------

Description

Add a representation and its parameters.

Usage

```
addRepresentation(NGLVieweR, type, param = list())
```

Arguments

NGLVieweR	A NGLVieweR object.
type	Type of representation. Most common options are "cartoon", "ball+stick", "line", "surface", "ribbon" and "label". For a full list of options, see the "structureRepresentation" method in the official NGL.js manual.
param	Options for the different types of representations. Most common options are name, opacity, colorScheme, sele, colorValue and visibility. For a full list of options, see the general "RepresentationParameters" method and type specific Label-, Structure- and Surface- RepresentationParameters in the official NGL.js manual.

Value

List of representation parameters to NGLVieweR htmlwidgets object.

See Also

- [addSelection\(\)](#)
- [NGLVieweR_example\(\)](#) See example "basic".

Examples

```

NGLVieweR("7CID") %>%
  stageParameters(backgroundColor = "black") %>%
  addRepresentation("cartoon", param = list(name = "cartoon", colorValue = "blue")) %>%
  addRepresentation("ball+stick", param = list(
    name = "ball+stick", sele = "241",
    colorScheme = "element", colorValue = "yellow"
  )) %>%
  addRepresentation("label",
    param = list(
      name = "label",
      showBackground = TRUE,
      labelType = "res",
      color = "black",
      backgroundColor = "white",
      backgroundOpacity = 0.8,
      sele = ":A and 241 and .CG"
    )
  )
)

# Shiny context
if (interactive()) {
  library(shiny)
  ui <- fluidPage(NGLVieweROutput("structure"))
  server <- function(input, output) {
    output$structure <- renderNGLVieweR({
      NGLVieweR("7CID") %>%
        stageParameters(backgroundColor = "black") %>%
        addRepresentation("cartoon",
          param = list(name = "cartoon", colorValue = "blue")
        ) %>%
        addRepresentation("ball+stick",
          param = list(
            name = "ball+stick", sele = "241",
            colorScheme = "element"
          )
        ) %>%
        addRepresentation("label",
          param = list(
            name = "label",
            showBackground = TRUE,
            labelType = "res",
            colorValue = "black",

```

```

        backgroundColor = "white",
        backgroundOpacity = 0.8,
        sele = ":A and 241 and .CG"
      )
    )
  })
}
shinyApp(ui, server)
}

```

 addSelection

Add a selection

Description

Add a new selection to a NGLVieweR object in Shiny mode.

Usage

```
addSelection(NGLVieweR_proxy, type, param = list())
```

Arguments

NGLVieweR_proxy	A NGLVieweR object.
type	Type of representation. Most common options are "cartoon", "ball+stick", "surface", "ribbon" and "label".
param	Options for the different types of representations. Most common options are name, opacity, colorScheme, sele, colorValue and visibility. For a full list of options, see the general "RepresentationParameters" method and type specific Label-, Structure- and Surface- RepresentationParameters in the official NGL.js manual.

Value

API call containing NGLVieweR id and list of message parameters.

See Also

- [updateRepresentation\(\)](#) Update an existing NGLVieweR representation.
- [NGLVieweR_example\(\)](#) See example "addSelection".

Other selections: [removeSelection\(\)](#), [updateSelection\(\)](#)

Examples

```

## Not run:
NGLViewer_proxy("7CID") %>%
  addSelection("ball+stick", param = list(name="sel1",
                                          sele="1-20",
                                          colorValue="yellow",
                                          colorScheme="element"
                                          ))

## End(Not run)

if (interactive()) {
  library(shiny)

  ui <- fluidPage(
    titlePanel("Viewer with API inputs"),
    sidebarLayout(
      sidebarPanel(
        textInput("selection", "Selection", "1-20"),
        selectInput("type", "Type", c("ball+stick", "cartoon", "backbone")),
        selectInput("color", "Color", c("orange", "grey", "white")),
        actionButton("add", "Add"),
        actionButton("remove", "Remove")
      ),
      mainPanel(
        NGLViewerOutput("structure")
      )
    )
  )
  server <- function(input, output) {
    output$structure <- renderNGLViewer({
      NGLViewer("7CID") %>%
        addRepresentation("cartoon",
                          param = list(name = "cartoon", colorScheme = "residueindex")
        )
    })
    observeEvent(input$add, {
      NGLViewer_proxy("structure") %>%
        addSelection(isolate(input$type),
                     param =
                       list(
                         name = "sel1",
                         sele = isolate(input$selection),
                         colorValue = isolate(input$color)
                       )
        )
    })
    observeEvent(input$remove, {
      NGLViewer_proxy("structure") %>%
        removeSelection("sel1")
    })
  }
}

```

```
}  
shinyApp(ui, server)  
}
```

NGLVieweR

Create a NGLVieweR

Description

NGLVieweR can be used to visualize and interact with Protein Data Bank (PDB) and structural files in R and Shiny applications. It includes a set of API functions to manipulate the viewer after creation in Shiny.

Usage

```
NGLVieweR(data, format = NULL, width = NULL, height = NULL, elementId = NULL)
```

Arguments

<code>data</code>	PDB file or PDB entry code
<code>format</code>	Input format (.mmCIF, .cif, .mcif, .pdb, .ent, .pqr, .gro, .sdf, .sd, .mol2, .mmTF). Needed when no file extension is provided.
<code>width, height</code>	Must be a valid CSS unit (like '100%', '400px', 'auto') or a number, which will be coerced to a string and have 'px' appended.
<code>elementId</code>	optional element Id

Details

The package is based on the [NGL.js](#) JavaScript library. To see the full set of features please read the official manual of [NGL.js](#).

Value

A NGLVieweR htmlwidgets object.

See Also

- [NGLVieweR_proxy\(\)](#) for handling of API calls after rendering.
- [NGLVieweR_example\(\)](#) See example "API" and "basic".

Examples

```

# Example 1: Most Basic
NGLVieweR("7CID") %>%
  addRepresentation("cartoon", param = list(name = "cartoon", colorScheme="residueindex"))

# Example 2: Advanced
NGLVieweR("7CID") %>%
  stageParameters(backgroundcolor = "white") %>%
  setQuality("high") %>%
  setSpin(FALSE) %>%
  addRepresentation("cartoon",
    param = list(
      name = "cartoon",
      colorScheme = "residueindex"
    )
  ) %>%
  addRepresentation("ball+stick",
    param = list(
      name = "ball+stick",
      colorValue = "red",
      colorScheme = "element",
      sele = "200"
    )
  ) %>%
  addRepresentation("label",
    param = list(
      name = "label", sele = "200:A.0",
      showBackground = TRUE,
      backgroundColor = "black",
      backgroundMargin = 2,
      backgroundOpacity = 0.5,
      showBorder = TRUE,
      colorValue = "white"
    )
  ) %>%
  addRepresentation("surface",
    param = list(
      name = "surface",
      colorValue = "white",
      opacity = 0.1
    )
  ) %>%
  zoomMove("200", "200", 2000, -20)

#-----Using Shiny-----

# App 1: Basic Example
if (interactive()) {
  library(shiny)
  ui <- fluidPage(NGLVieweROutput("structure"))
  server <- function(input, output) {
    output$structure <- renderNGLVieweR({

```

```

NGLVieweR("7CID") %>%
  addRepresentation("cartoon",
    param = list(
      name = "cartoon",
      colorScheme = "residueindex"
    )
  ) %>%
  addRepresentation("ball+stick",
    param = list(
      name = "cartoon",
      sele = "1-20",
      colorScheme = "element"
    )
  ) %>%
  stageParameters(backgroundColor = "black") %>%
  setQuality("high") %>%
  setFocus(0) %>%
  setSpin(TRUE)
})
}
shinyApp(ui, server)
}

# App 2: Example with API calls
if (interactive()) {
library(shiny)

ui <- fluidPage(
  titlePanel("Viewer with API inputs"),
  sidebarLayout(
    sidebarPanel(
      textInput("selection", "Selection", "1-20"),
      selectInput("type", "Type", c("ball+stick", "cartoon", "backbone")),
      selectInput("color", "Color", c("orange", "grey", "white")),
      actionButton("add", "Add"),
      actionButton("remove", "Remove")
    ),
    mainPanel(
      NGLVieweROutput("structure")
    )
  )
)

server <- function(input, output) {
  output$structure <- renderNGLVieweR({
    NGLVieweR("7CID") %>%
      addRepresentation("cartoon",
        param = list(name = "cartoon", colorScheme = "residueindex")
      ) %>%
      stageParameters(backgroundColor = input$backgroundColor) %>%
      setQuality("high") %>%
      setFocus(0) %>%
      setSpin(TRUE)
  })
}
}

```



```

observeEvent(input$add, {
  NGLVieweR_proxy("structure") %>%
  addSelection(isolate(input$type),
    param =
      list(
        name = "sel1",
        sele = isolate(input$selection),
        colorValue = isolate(input$color)
      )
  )
})

observeEvent(input$remove, {
  NGLVieweR_proxy("structure") %>%
  removeSelection("sel1")
})
}
shinyApp(ui, server)
}

```

NGLVieweR-shiny

Shiny bindings for NGLVieweR

Description

Output and render functions for using NGLVieweR within Shiny applications and interactive Rmd documents.

Usage

```
NGLVieweROutput(outputId, width = "100%", height = "400px")
```

```
renderNGLVieweR(expr, env = parent.frame(), quoted = FALSE)
```

```
NGLVieweR_proxy(id, session = shiny::getDefaultReactiveDomain())
```

Arguments

outputId	output variable to read from
width, height	Must be a valid CSS unit (like '100%', '400px', 'auto') or a number, which will be coerced to a string and have 'px' appended.
expr	An expression that generates a NGLVieweR.
env	The environment in which to evaluate expr.
quoted	Is expr a quoted expression (with quote())? This is useful if you want to save an expression in a variable.
id	single-element character vector indicating the output ID of the chart to modify (if invoked from a Shiny module, the namespace will be added automatically)

`session` The Shiny session object to which the map belongs; usually the default value will suffice.

Value

NGLVieweR object that can be placed in the UI.

See Also

[NGLVieweR_example\(\)](#)

Examples

```
if (interactive()) {
  library(shiny)

  ui <- fluidPage(
    titlePanel("Viewer with API inputs"),
    sidebarLayout(
      sidebarPanel(
        textInput("selection", "Selection", "1-20"),
        selectInput("type", "Type", c("ball+stick", "cartoon", "backbone")),
        selectInput("color", "Color", c("orange", "grey", "white")),
        actionButton("add", "Add"),
        actionButton("remove", "Remove")
      ),
      mainPanel(
        NGLVieweROutput("structure")
      )
    )
  )
  server <- function(input, output) {
    output$structure <- renderNGLVieweR({
      NGLVieweR("7CID") %>%
        addRepresentation("cartoon",
          param = list(name = "cartoon", color = "residueindex")
        ) %>%
        stageParameters(backgroundColor = input$backgroundColor) %>%
        setQuality("high") %>%
        setFocus(0) %>%
        setSpin(TRUE)
    })
    observeEvent(input$add, {
      NGLVieweR_proxy("structure") %>%
        addSelection(isolate(input$type),
          param =
            list(
              name = "sel1",
              sele = isolate(input$selection),
              color = isolate(input$color)
            )
        )
    })
  }
}
```

```
  })
  observeEvent(input$remove, {
    NGLVieweR_proxy("structure") %>%
      removeSelection("sel1")
  })
}
shinyApp(ui, server)
}
```

NGLVieweR_example *Run NGLVieweR example Shiny app*

Description

Launch an example to demonstrate how to use NGLvieweR in Shiny.

Usage

```
NGLVieweR_example(example = "basic")
```

Arguments

example	Example type for which to see an example, possible values are: "basic", "API", "addSelection", "removeSelection", "snapshot", "updateAnimation", "updateColor", "updateFocus", "updateFullscreen", "updateRepresentation", "updateSelection", "updateStage", "updateVisibility" and "updateZoomMove".
---------	---

Value

Call to load Shiny example.

Examples

```
if (interactive()) {
  # Basic example
  NGLVieweR_example("basic")

  # Example with API calls
  NGLVieweR_example("API")

  # Function specific example
  NGLVieweR_example("addSelection")
}
```

removeSelection	<i>Remove a selection</i>
-----------------	---------------------------

Description

Remove an existing NGLVieweR selection in Shiny mode.

Usage

```
removeSelection(NGLVieweR_proxy, name)
```

Arguments

NGLVieweR_proxy	A NGLVieweR object.
name	Name of selection to be removed.

Value

API call containing NGLVieweR id and list of message parameters.

See Also

- [NGLVieweR_example\(\)](#) See example "removeSelection".

Other selections: [addSelection\(\)](#), [updateSelection\(\)](#)

Examples

```
## Not run:
NGLVieweR_proxy("structure") %>%
  removeSelection("sel1")

## End(Not run)

if (interactive()) {
  library(shiny)

  ui <- fluidPage(
    titlePanel("Viewer with API inputs"),
    sidebarLayout(
      sidebarPanel(
        textInput("selection", "Selection", "1-20"),
        selectInput("type", "Type", c("ball+stick", "cartoon", "backbone")),
        selectInput("color", "Color", c("orange", "grey", "white")),
        actionButton("add", "Add"),
        actionButton("remove", "Remove")
      ),
      mainPanel(
        NGLVieweROutput("structure")
      )
    )
  }
}
```

```

    )
  )
)
server <- function(input, output) {
  output$structure <- renderNGLViewR({
    NGLViewR("7CID") %>%
      addRepresentation("cartoon",
        param = list(name = "cartoon", colorScheme = "residueindex")
      )
  })
  observeEvent(input$add, {
    NGLViewR_proxy("structure") %>%
      addSelection(isolate(input$type),
        param =
          list(
            name = "sel1",
            sele = isolate(input$selection),
            colorValue = isolate(input$color)
          )
      )
  })
  observeEvent(input$remove, {
    NGLViewR_proxy("structure") %>%
      removeSelection("sel1")
  })
}
shinyApp(ui, server)
}

```

selectionParameters *Set selection parameters*

Description

Set selection parameters.

Usage

```
selectionParameters(NGLViewR, proximity = 3, level = "residue")
```

Arguments

NGLViewR	A NGLViewR object.
proximity	Set distance in angstrom for atoms to return in proximity of selection. Default = 3.
level	Set level on which atoms in proximity of selection are returned. Options are "residue" (default) or "atom".

Value

Returns list of selection parameters to NGLViewer htmlwidgets object.

Examples

```
NGLViewer("7CID") %>%
  addRepresentation("cartoon") %>%
  selectionParameters(3, "residue")

# Shiny context
if (interactive()) {
  library(shiny)
  ui <- fluidPage(NGLViewerOutput("structure"))
  server <- function(input, output) {
    output$structure <- renderNGLViewer({
      NGLViewer("7CID") %>%
        addRepresentation("cartoon") %>%
        selectionParameters(3, "residue")
    })
    observeEvent(input$structure_selAround, {
      NGLViewer_proxy("structure") %>% removeSelection("selAround")
      NGLViewer_proxy("structure") %>%
        addSelection(
          "ball+stick",
          param =
            list(
              name = "selAround",
              sele = input$structure_selAround,
              colorValue = "grey"
            )
        )
    })
  }
  shinyApp(ui, server)
}
```

setFocus

Set Focus

Description

Set Focus

Usage

```
setFocus(NGLViewer, focus = 0)
```

Arguments

NGLVieweR	A NGLVieweR object.
focus	Set focus between 0 (default) to 100.

Value

setFocus parameter in NGLVieweR htmlwidgets object.

See Also

[updateFocus\(\)](#)

Other options: [setQuality\(\)](#), [snapShot\(\)](#), [updateFocus\(\)](#), [updateFullscreen\(\)](#)

Examples

```
NGLVieweR("7CID") %>%  
  addRepresentation("cartoon", param=list(name="cartoon", colorValue="blue")) %>%  
  setFocus(0)
```

setPosition	<i>Set Position</i>
-------------	---------------------

Description

Set position for the representation

Usage

```
setPosition(NGLVieweR, x = 0, y = 0, z = 0)
```

Arguments

NGLVieweR	A NGLVieweR object.
x	Position along the x-axis in angstroms. Default is 0.
y	Position along the y-axis in angstroms. Default is 0.
z	Position along the z-axis in angstroms. Default is 0.

Value

NGLVieweR object with updated setPosition parameters.

See Also

- [setScale\(\)](#)
- [zoomMove\(\)](#)
- [setRotation\(\)](#)

Other transformations: [setRotation\(\)](#), [setScale\(\)](#), [zoomMove\(\)](#)

Examples

```
NGLVieweR("7CID") %>%
stageParameters(backgroundColor = "white") %>%
addRepresentation("cartoon", param=list(name="cartoon", colorValue="red")) %>%
addRepresentation("ball+stick", param=list(name="ball+stick",
colorValue="yellow",
colorScheme="element",
sele="200")) %>%
setPosition(25, 0, 0)
```

setQuality

Set Quality

Description

Set Quality

Usage

```
setQuality(NGLVieweR, quality = "medium")
```

Arguments

NGLVieweR	A NGLVieweR object.
quality	Set rendering quality. Can be "low", "medium" (default) or "high".

Value

setQuality parameter in NGLVieweR htmlwidgets object.

See Also

Other options: [setFocus\(\)](#), [snapShot\(\)](#), [updateFocus\(\)](#), [updateFullscreen\(\)](#)

Examples

```
NGLVieweR("7CID") %>%
addRepresentation("cartoon", param=list(name="cartoon", colorValue="blue")) %>%
setQuality("medium")
```

setRock	<i>Set rock</i>
---------	-----------------

Description

Set rock animation

Usage

```
setRock(NGLVieweR, rock = TRUE)
```

Arguments

NGLVieweR	A NGLVieweR object.
rock	If TRUE (default), start rocking and stop spinning.

Value

setRock parameter to TRUE or FALSE in NGLVieweR htmlwidgets object.

See Also

- [setSpin\(\)](#)
- [updateRock\(\)](#)

Other animations: [setSpin\(\)](#), [updateRock\(\)](#), [updateSpin\(\)](#), [updateZoomMove\(\)](#)

Examples

```
NGLVieweR("7CID") %>%  
  addRepresentation("cartoon", param=list(name="cartoon", colorValue="blue")) %>%  
  setRock(TRUE)
```

setRotation	<i>Rotate View</i>
-------------	--------------------

Description

Set rotation for the representation

Usage

```
setRotation(NGLVieweR, x = 0, y = 0, z = 0, degrees = TRUE)
```

Arguments

NGLVieweR	A NGLVieweR object.
x	Rotation angle around the x-axis. Default is 0.
y	Rotation angle around the y-axis. Default is 0.
z	Rotation angle around the z-axis. Default is 0.
degrees	A logical value. If TRUE (default), the input angles are assumed to be in degrees and will be converted to radians. If FALSE, the angles are assumed to be in radians.

Value

NGLVieweR object with updated rotateView parameters.

See Also

- [setScale\(\)](#)
- [zoomMove\(\)](#)
- [setPosition\(\)](#)

Other transformations: [setPosition\(\)](#), [setScale\(\)](#), [zoomMove\(\)](#)

Examples

```
NGLVieweR("7CID") %>%
stageParameters(backgroundColor = "white") %>%
addRepresentation("cartoon", param=list(name="cartoon", colorValue="red")) %>%
addRepresentation("ball+stick", param=list(name="ball+stick",
colorValue="yellow",
colorScheme="element",
sele="200")) %>%
setRotation(30, 45, 60, degrees = TRUE)
```

setScale

Set Scale

Description

Set the scale factor for the representation

Usage

```
setScale(NGLVieweR, scale = 1)
```

Arguments

NGLVieweR	A NGLVieweR object.
scale	A numeric value indicating the scale factor (default is 1).

Value

Updated NGLVieweR object with new scale parameter.

See Also

- [zoomMove\(\)](#)
- [setRotation\(\)](#)
- [setPosition\(\)](#)

Other transformations: [setPosition\(\)](#), [setRotation\(\)](#), [zoomMove\(\)](#)

Examples

```
NGLVieweR("7CID") %>%  
addRepresentation("cartoon", param=list(name="cartoon", colorValue="blue")) %>%  
setScale(2)
```

setSpin

Set Spin

Description

Set Spin animation

Usage

```
setSpin(NGLVieweR, spin = TRUE)
```

Arguments

NGLVieweR	A NGLVieweR object.
spin	If TRUE (default), start spinning and stop rocking

Value

setSpin parameter to TRUE or FALSE in NGLVieweR htmlwidgets object.

See Also

- [setRock\(\)](#)
- [updateSpin\(\)](#)

Other animations: [setRock\(\)](#), [updateRock\(\)](#), [updateSpin\(\)](#), [updateZoomMove\(\)](#)

Examples

```
NGLVieweR("7CID") %>%  
addRepresentation("cartoon", param=list(name="cartoon", colorValue="blue")) %>%  
setSpin(TRUE)
```

`snapShot`*Snapshot*

Description

Make a snapshot of a NGLVieweR object in Shiny mode.

Usage

```
snapShot(NGLVieweR_proxy, fileName = "Snapshot", param = list())
```

Arguments

<code>NGLVieweR_proxy</code>	A NGLVieweR object.
<code>fileName</code>	Optional name for Snapshot (default = "Snapshot").
<code>param</code>	Of type list, can be; antialias TRUE/FALSE, trim TRUE/FALSE, transparent TRUE/FALSE or scale numeric. For a full list of options, see "makeImage" and "ImageParameters" in the official NGL.js manual.

Value

API call containing NGLVieweR id and list of message parameters.

See Also

[NGLVieweR_example\(\)](#) See example "snapshot".

Other options: [setFocus\(\)](#), [setQuality\(\)](#), [updateFocus\(\)](#), [updateFullscreen\(\)](#)

Examples

```
## Not run:
NGLVieweR_proxy("structure") %>%
snapShot("Snapshot", param = list(
  antialias = TRUE,
  trim = TRUE,
  transparent = TRUE,
  scale = 1))

## End(Not run)

if (interactive()) {
  library(shiny)

  ui <- fluidPage(
    titlePanel("Viewer with API inputs"),
    sidebarLayout(
      sidebarPanel(
```

```

        actionButton("snapshot", "Snapshot"),
      ),
      mainPanel(
        NGLViewerOutput("structure")
      )
    )
  )
  server <- function(input, output) {
    output$structure <- renderNGLViewer({
      NGLViewer("7CID") %>%
        addRepresentation("cartoon",
          param = list(
            name = "cartoon",
            color = "residueindex"
          )
        )
    })
    observeEvent(input$snapshot, {
      NGLViewer_proxy("structure") %>%
        snapShot("Snapshot",
          param = list(
            antialias = TRUE,
            trim = TRUE,
            transparent = TRUE,
            scale = 1
          )
        )
    })
  }
  shinyApp(ui, server)
}

```

stageParameters	<i>Set stage parameters</i>
-----------------	-----------------------------

Description

Set stage parameters.

Usage

```
stageParameters(NGLViewer, ...)
```

Arguments

NGLViewer	A NGLViewer object.
...	Options controlling the stage. Most common options are <code>backgroundColor</code> , <code>rotateSpeed</code> , <code>zoomSpeed</code> , <code>hoverTimeout</code> and <code>lightIntensity</code> . For a full list of options, see the "stageParameters" method in the official NGL.js manual.

Value

Returns list of stage parameters to NGLVieweR htmlwidgets object.

See Also

- [updateStage\(\)](#)
- [NGLVieweR_example\(\)](#) See example "basic".

Examples

```
NGLVieweR("7CID") %>%
  stageParameters(backgroundColor = "white", zoomSpeed = 1) %>%
  addRepresentation("cartoon", param = list(name = "cartoon", colorScheme="residueindex"))

if (interactive()) {
  library(shiny)
  ui <- fluidPage(NGLVieweROutput("structure"))
  server <- function(input, output) {
    output$structure <- renderNGLVieweR({
      NGLVieweR("7CID") %>%
        stageParameters(backgroundColor = "white", zoomSpeed = 1) %>%
        addRepresentation("cartoon",
          param = list(name = "cartoon", colorScheme = "residueindex")
        )
    })
  }
  shinyApp(ui, server)
}
```

updateColor

Update color of a selection

Description

Update color of an existing NGLVieweR selection in Shiny mode.

Usage

```
updateColor(NGLVieweR_proxy, name, color)
```

Arguments

NGLVieweR_proxy	A NGLVieweR object.
name	Name of selection to alter the color.
color	Can be a colorValue (color name or HEX code) or colorScheme (e.g. "element", "resname", "random" or "residueindex"). For a full list of options, see the "Colormaker" section in the official NGL.js manual.

Value

API call containing NGLViewer id and list of message parameters.

See Also

- [NGLViewer_example\(\)](#) See example "updateColor".

Other updates: [updateRepresentation\(\)](#), [updateStage\(\)](#), [updateVisibility\(\)](#)

Examples

```
## Not run:
NGLViewer_proxy("structure") %>%
  updateColor("cartoon", "red")

## End(Not run)

if (interactive()) {
  library(shiny)

  ui <- fluidPage(
    titlePanel("Viewer with API inputs"),
    sidebarLayout(
      sidebarPanel(
        colourInput("color", "red", "red"),
        actionButton("update", "Update"),
      ),
      mainPanel(
        NGLViewerOutput("structure")
      )
    )
  )
  server <- function(input, output) {
    output$structure <- renderNGLViewer({
      NGLViewer("7CID") %>%
        addRepresentation("cartoon",
          param = list(name = "cartoon", color = "residueindex")
        )
    })
    observeEvent(input$update, {
      NGLViewer_proxy("structure") %>%
        updateColor("cartoon", isolate(input$color))
    })
  }
  shinyApp(ui, server)
}
```

`updateFocus`*Update Focus*

Description

Update the focus of an existing NGLVieweR object in Shiny mode.

Usage

```
updateFocus(NGLVieweR_proxy, focus = 0)
```

Arguments

<code>NGLVieweR_proxy</code>	A NGLVieweR object.
<code>focus</code>	Numeric value between 0-100 (default = 0).

Value

API call containing NGLVieweR id and list of message parameters.

See Also

- [setFocus\(\)](#)
- [NGLVieweR_example\(\)](#) See example "updateFocus".

Other options: [setFocus\(\)](#), [setQuality\(\)](#), [snapShot\(\)](#), [updateFullscreen\(\)](#)

Examples

```
## Not run:
NGLVieweR_proxy("structure") %>%
  updateFocus(focus = 50)

## End(Not run)

if (interactive()) {
  library(shiny)
  ui = fluidPage(
    titlePanel("Viewer with API inputs"),
    sidebarLayout(
      sidebarPanel(
        sliderInput("focus", "Focus", 0, 100, 50)
      ),
      mainPanel(
        NGLVieweROutput("structure")
      )
    )
  )
}
```



```

server = function(input, output) {
  output$structure <- renderNGLViewer({
    NGLViewer("7CID") %>%
      addRepresentation("cartoon",
        param = list(name = "cartoon", color= "red"))
  })
  observeEvent(input$focus, {
    NGLViewer_proxy("structure") %>%
      updateFocus(input$focus)
  })
}
shinyApp(ui, server)
}

```

updateFullscreen	<i>Fullscreen</i>
------------------	-------------------

Description

Put viewer in fullscreen. Works in Shiny mode.

Usage

```
updateFullscreen(NGLViewer_proxy, fullscreen = TRUE)
```

Arguments

NGLViewer_proxy	A NGLViewer object.
fullscreen	If TRUE put viewer in fullscreen.

Value

API call containing NGLViewer id and list of message parameters.

See Also

[NGLViewer_example\(\)](#) See example "updateFullscreen".
 Other options: [setFocus\(\)](#), [setQuality\(\)](#), [snapShot\(\)](#), [updateFocus\(\)](#)

Examples

```

## Not run:
NGLViewer_proxy("structure") %>% updateFullscreen()

## End(Not run)

if (interactive()) {
  library(shiny)

```

```

ui <- fluidPage(
  titlePanel("Viewer with API inputs"),
  sidebarLayout(
    sidebarPanel(
      actionButton("fullscreen", "Fullscreen"),
    ),
    mainPanel(
      NGLViewerOutput("structure")
    )
  )
)
server = function(input, output) {
  output$structure <- renderNGLViewer({
    NGLViewer("7CID") %>%
      addRepresentation("cartoon",
        param = list(name = "cartoon", color = "red")
      )
  })

  observeEvent(input$fullscreen, {
    NGLViewer_proxy("structure") %>%
      updateFullscreen()
  })
}
shinyApp(ui, server)
}

```

updateRepresentation *Update Representation*

Description

Update an existing NGLViewer representation in Shiny mode.

Usage

```
updateRepresentation(NGLViewer_proxy, name, param = list())
```

Arguments

NGLViewer_proxy	A NGLViewer object.
name	Name of representation to alter the color.
param	Options for the different types of representations. Most common options are name, opacity, colorScheme, colorValue and visibility. For a full list of options, see the general "RepresentationParameters" method and type specific Label-, Structure- and Surface- RepresentationParameters in the official NGL.js manual.

Value

API call containing NGLViewer id and list of message parameters.

See Also

- [addSelection\(\)](#) Add a new selection to a NGLViewer object.
- [addRepresentation\(\)](#)
- [NGLViewer_example\(\)](#) See example "updateRepresentation".

Other updates: [updateColor\(\)](#), [updateStage\(\)](#), [updateVisibility\(\)](#)

Examples

```
## Not run:
NGLViewer_proxy("structure") %>%
  updateRepresentation("cartoon",
    param = list(
      name = "cartoon",
      color = isolate(input$color),
      opacity = isolate(input$opacity)
    )
  )

## End(Not run)

if (interactive()) {
  library(shiny)

  ui = fluidPage(
    titlePanel("Viewer with API inputs"),
    sidebarLayout(
      sidebarPanel(
        selectInput("color", "Color", c("red", "white", "blue")),
        sliderInput("opacity", "Opacity", 0, 1, 1),
        actionButton("update", "Update"),
      ),
      mainPanel(
        NGLViewerOutput("structure")
      )
    )
  )
  server = function(input, output) {
    output$structure <- renderNGLViewer({
      NGLViewer("7CID") %>%
        addRepresentation("cartoon",
          param = list(name = "cartoon", color="red"))
    })
    observeEvent(input$update, {
      NGLViewer_proxy("structure") %>%
        updateRepresentation("cartoon",
          param = list(
            color = isolate(input$color),
```

```

        opacity = isolate(input$opacity)
      )
    )
  })
}
shinyApp(ui, server)
}

```

updateRock

Update Rock

Description

Start rock animation and stop spinning. Works on an existing NGLVieweR object in Shiny mode.

Usage

```
updateRock(NGLVieweR_proxy, rock = TRUE)
```

Arguments

NGLVieweR_proxy

A NGLVieweR object.

rock

If TRUE (default), start rocking and stop spinning.

Value

API call containing NGLVieweR id and list of message parameters.

See Also

- [setRock\(\)](#)
- [NGLVieweR_example\(\)](#) See example "updateAnimation".

Other animations: [setRock\(\)](#), [setSpin\(\)](#), [updateSpin\(\)](#), [updateZoomMove\(\)](#)

Examples

```

## Not run:
NGLVieweR_proxy("structure") %>% updateRock(TRUE)

## End(Not run)

if (interactive()) {
  library(shiny)

  ui = fluidPage(
    titlePanel("Viewer with API inputs"),
    sidebarLayout(

```

```

    sidebarPanel(
      radioButtons("animate", label = "Animation",
        choices = c("None", "Spin", "Rock"), selected = "None")
    ),
    mainPanel(
      NGLViewROutput("structure")
    )
  )
)
server = function(input, output) {
  output$structure <- renderNGLViewR({
    NGLViewR("7CID") %>%
      addRepresentation("cartoon",
        param = list(name = "cartoon", color="red"))
  })

  observeEvent(input$animate,{
    if(input$animate == "Rock"){
      NGLViewR_proxy("structure") %>%
        updateRock(TRUE)
    } else if(input$animate == "Spin") {
      NGLViewR_proxy("structure") %>%
        updateSpin(TRUE)
    } else{
      NGLViewR_proxy("structure") %>%
        updateRock(FALSE) %>%
        updateSpin(FALSE)
    }
  })
}
shinyApp(ui, server)
}

```

updateSelection

Update a selection

Description

Update the selected residues of an existing NGLViewR selection in

Usage

```
updateSelection(NGLViewR_proxy, name = name, sele = "none")
```

Arguments

NGLViewR_proxy	A NGLViewR object.
name	Name of selection.
sele	Selected atoms/residues. See the section "selection-language" in the official NGL.js manual.

Value

API call containing NGLViewer id and list of message parameters.

See Also

- [NGLViewer_example\(\)](#) See example "updateSelection".

Other selections: [addSelection\(\)](#), [removeSelection\(\)](#)

Examples

```
## Not run:
NGLViewer_proxy("structure") %>%
  updateSelection("ball+stick", sele = "1-20")

## End(Not run)

if (interactive()) {
  library(shiny)
  ui <- fluidPage(
    titlePanel("Viewer with API inputs"),
    sidebarLayout(
      sidebarPanel(
        textInput("selection", "Selection", "1-20"),
        actionButton("update", "Update")
      ),
      mainPanel(
        NGLViewerOutput("structure")
      )
    )
  )
  server <- function(input, output) {
    output$structure <- renderNGLViewer({
      NGLViewer("7CID") %>%
        addRepresentation("cartoon",
          param = list(name = "cartoon", color = "red")
        ) %>%
        addRepresentation("ball+stick",
          param = list(
            name = "ball+stick",
            colorValue = "yellow",
            colorScheme = "element",
            sele = "1-20"
          )
        )
    })
    observeEvent(input$update, {
      NGLViewer_proxy("structure") %>%
        updateSelection("ball+stick", sele = isolate(input$selection))
    })
  }
  shinyApp(ui, server)
}
```

updateSpin	<i>Update Spin</i>
------------	--------------------

Description

Start spin animation and stop rocking. Works on an existing NGLVieweR object in Shiny mode.

Usage

```
updateSpin(NGLVieweR_proxy, spin = TRUE)
```

Arguments

NGLVieweR_proxy	A NGLVieweR object.
spin	If TRUE (default), start spinning and stop rocking.

Value

API call containing NGLVieweR id and list of message parameters.

See Also

- [setSpin\(\)](#)
- [NGLVieweR_example\(\)](#) See example "updateAnimation".

Other animations: [setRock\(\)](#), [setSpin\(\)](#), [updateRock\(\)](#), [updateZoomMove\(\)](#)

Examples

```
## Not run:
NGLVieweR_proxy("structure") %>% updateRock(TRUE)

## End(Not run)
if (interactive()) {
  library(shiny)

  ui = fluidPage(
    titlePanel("Viewer with API inputs"),
    sidebarLayout(
      sidebarPanel(
        radioButtons("animate", label = "Animation",
                    choices = c("None", "Spin", "Rock"), selected = "None")
      ),
      mainPanel(
        NGLVieweROutput("structure")
      )
    )
  )
}
```

```

server = function(input, output) {
  output$structure <- renderNGLViewer({
    NGLViewer("7CID") %>%
      addRepresentation("cartoon",
        param = list(name = "cartoon", color="red"))
  })

  observeEvent(input$animate,{
    if(input$animate == "Rock"){
      NGLViewer_proxy("structure") %>%
        updateRock(TRUE)
    } else if(input$animate == "Spin") {
      NGLViewer_proxy("structure") %>%
        updateSpin(TRUE)
    } else{
      NGLViewer_proxy("structure") %>%
        updateRock(FALSE) %>%
        updateSpin(FALSE)
    }
  })
}
shinyApp(ui, server)
}

```

updateStage

Update Stage

Description

Update an existing NGLViewer stage in Shiny mode.

Usage

```
updateStage(NGLViewer_proxy, param = list())
```

Arguments

NGLViewer_proxy	A NGLViewer object.
param	Of type list. Most common options are backgroundColor, rotateSpeed, zoomSpeed, hoverTimeout and lightIntensity. For a full list of options, see the "StageParameters" method in the official NGL.js manual.

Value

API call containing NGLViewer id and list of message parameters.

See Also

- [stageParameters\(\)](#)
- [NGLViewer_example\(\)](#) See example "updateStage".

Other updates: [updateColor\(\)](#), [updateRepresentation\(\)](#), [updateVisibility\(\)](#)

Examples

```
## Not run:
NGLViewer("7CID") %>%
  addRepresentation("cartoon",
                    param = list(name = "cartoon", color="red")) %>%
  stageParameters(background = "black")

## End(Not run)

if (interactive()) {
  library(shiny)

  ui = fluidPage(
    titlePanel("Viewer with API inputs"),
    sidebarLayout(
      sidebarPanel(
        selectInput("background", "Background", c("black", "white", "blue")),
        actionButton("update", "Update"),
      ),
      mainPanel(
        NGLViewerOutput("structure")
      )
    )
  )
  server <- function(input, output) {
    output$structure <- renderNGLViewer({
      NGLViewer("7CID") %>%
        addRepresentation("cartoon",
                          param = list(name = "cartoon", color = "red")
        ) %>%
        stageParameters(background = "black")
    })
    observeEvent(input$update, {
      NGLViewer_proxy("structure") %>%
        updateStage(
          param = list("backgroundColor" = isolate(input$background))
        )
    })
  }
  shinyApp(ui, server)
}
```

updateVisibility	<i>Update visibility</i>
------------------	--------------------------

Description

Hide or show an existing NGLVieweR selection in Shiny mode.

Usage

```
updateVisibility(NGLVieweR_proxy, name, value = FALSE)
```

Arguments

NGLVieweR_proxy	A NGLVieweR object.
name	Name of selection to alter the color.
value	Hide FALSE or show TRUE selection. For a full description see "setVisibility" in the official NGL.js manual.

Value

API call containing NGLVieweR id and list of message parameters.

See Also

[NGLVieweR_example\(\)](#) See example "updateVisibility".

Other updates: [updateColor\(\)](#), [updateRepresentation\(\)](#), [updateStage\(\)](#)

Examples

```
## Not run:
NGLVieweR_proxy("structure") %>%
  updateVisibility("cartoon", value = TRUE)

## End(Not run)

if (interactive()) {
  library(shiny)

  ui = fluidPage(
    titlePanel("Viewer with API inputs"),
    sidebarLayout(
      sidebarPanel(
        actionButton("show", "Show"),
        actionButton("hide", "Hide"),
      ),
      mainPanel(
        NGLVieweROutput("structure")
      )
    )
  )
}
```

```

    )
  )
)
server = function(input, output) {
  output$structure <- renderNGLViewer({
    NGLViewer("7CID") %>%
      addRepresentation("cartoon",
        param = list(name = "cartoon", color="residueindex"))
  })
  observeEvent(input$show, {
    NGLViewer_proxy("structure") %>%
      updateVisibility("cartoon", value = TRUE)
  })
  observeEvent(input$hide, {
    NGLViewer_proxy("structure") %>%
      updateVisibility("cartoon", value = FALSE)
  })
}
shinyApp(ui, server)
}

```

updateZoomMove

Update zoomMove

Description

Add a zoom animation on an existing NGLViewer object.

Usage

```
updateZoomMove(NGLViewer_proxy, center, zoom, duration = 0, z_offSet = 0)
```

Arguments

NGLViewer_proxy	A NGLViewer object.
center	Target distance of selected atoms/residues. See the section "selection-language" in the official NGL.js manual.
zoom	Target zoom of selected atoms/residues. See the section "selection-language" in the official NGL.js manual.
duration	Optional animation time in milliseconds (default = 0).
z_offSet	Optional zoom offset value (default = 0).

Value

API call containing NGLViewer id and list of message parameters.

See Also

- [zoomMove\(\)](#)
- [NGLViewerR_example\(\)](#) See example "updatezoomMove".

Other animations: [setRock\(\)](#), [setSpin\(\)](#), [updateRock\(\)](#), [updateSpin\(\)](#)

Examples

```
## Not run:
NGLViewerR_proxy("structure") %>% updateZoomMove(center = "200",
                                                  zoom = "200",
                                                  z_offSet = 80,
                                                  duration = 2000)

## End(Not run)

if (interactive()) {
library(shiny)

ui = fluidPage(
  titlePanel("Viewer with API inputs"),
  sidebarLayout(
    sidebarPanel(
      textInput("center", "Center", "200"),
      textInput("zoom", "Zoom", "200"),
      numericInput("zoomOffset", "Zoom offset", 80,0,100),
      numericInput("duration", "Duration", 2000,0,2000),
      actionButton("zoom", "Zoom"),
      actionButton("reset", "Reset")
    ),
    mainPanel(
      NGLViewerROutput("structure")
    )
  )
)

server = function(input, output) {
  output$structure <- renderNGLViewerR({
    NGLViewerR("7CID") %>%
      addRepresentation("cartoon",
        param = list(name = "cartoon", color="red")) %>%
      addRepresentation("ball+stick",
        param = list(name = "ball+stick", sele="200"))
  })

  observeEvent(input$zoom, {
    NGLViewerR_proxy("structure") %>%
      updateZoomMove(
        center = isolate(input$center),
        zoom = isolate(input$zoom),
        z_offSet = isolate(input$zoomOffset),
        duration = isolate(input$duration)
      )
  })
}
```

```

  })
  observeEvent(input$reset, {
    NGLVieweR_proxy("structure") %>%
      updateZoomMove(
        center = "*",
        zoom = "*",
        z_offSet = 0,
        duration = 1000
      )
  })
}
shinyApp(ui, server)
}

```

 zoomMove

Set zoomMove

Description

Add a zoom animation

Usage

```
zoomMove(NGLVieweR, center, zoom, duration = 0, z_offSet = 0)
```

Arguments

NGLVieweR	A NGLVieweR object.
center	Target distance of selected atoms/residues. See the section "selection-language" in the official NGL.js manual.
zoom	Target zoom of selected atoms/residues. See the section "selection-language" in the official NGL.js manual.
duration	Optional animation time in milliseconds (default = 0).
z_offSet	Optional zoom offset value (default = 0).

Value

List of zoomMove parameters to NGLVieweR htmlwidgets object.

See Also

- [setScale\(\)](#)
- [setRotation\(\)](#)
- [setPosition\(\)](#)

Other transformations: [setPosition\(\)](#), [setRotation\(\)](#), [setScale\(\)](#)

Examples

```
NGLViewer("7CID") %>%
stageParameters(backgroundColor = "white") %>%
  addRepresentation("cartoon", param=list(name="cartoon", colorValue="red")) %>%
  addRepresentation("ball+stick", param=list(name="ball+stick",
                                             colorValue="yellow",
                                             colorScheme="element",
                                             sele="200")) %>%
  zoomMove("200:A.C", "200:A.C", 2000, -20)
```

Index

- * **animations**
 - setRock, 17
 - setSpin, 19
 - updateRock, 28
 - updateSpin, 31
 - updateZoomMove, 35
- * **options**
 - setFocus, 14
 - setQuality, 16
 - snapShot, 20
 - updateFocus, 24
 - updateFullscreen, 25
- * **selections**
 - addSelection, 4
 - removeSelection, 12
 - updateSelection, 29
- * **transformations**
 - setPosition, 15
 - setRotation, 17
 - setScale, 18
 - zoomMove, 37
- * **updates**
 - updateColor, 22
 - updateRepresentation, 26
 - updateStage, 32
 - updateVisibility, 34
- addRepresentation, 2
- addRepresentation(), 27
- addSelection, 4, 12, 30
- addSelection(), 3, 27
- NGLViewer, 6
- NGLViewer-shiny, 9
- NGLViewer_example, 11
- NGLViewer_example(), 3, 4, 6, 10, 12, 20, 22–25, 27, 28, 30, 31, 33, 34, 36
- NGLViewer_proxy (NGLViewer-shiny), 9
- NGLViewer_proxy(), 6
- NGLViewerOutput (NGLViewer-shiny), 9
- removeSelection, 4, 12, 30
- renderNGLViewer (NGLViewer-shiny), 9
- selectionParameters, 13
- setFocus, 14, 16, 20, 24, 25
- setFocus(), 24
- setPosition, 15, 18, 19, 37
- setPosition(), 18, 19, 37
- setQuality, 15, 16, 20, 24, 25
- setRock, 17, 19, 28, 31, 36
- setRock(), 19, 28
- setRotation, 15, 17, 19, 37
- setRotation(), 15, 19, 37
- setScale, 15, 18, 18, 37
- setScale(), 15, 18, 37
- setSpin, 17, 19, 28, 31, 36
- setSpin(), 17, 31
- snapShot, 15, 16, 20, 24, 25
- stageParameters, 21
- stageParameters(), 33
- updateColor, 22, 27, 33, 34
- updateFocus, 15, 16, 20, 24, 25
- updateFocus(), 15
- updateFullscreen, 15, 16, 20, 24, 25
- updateRepresentation, 23, 26, 33, 34
- updateRepresentation(), 4
- updateRock, 17, 19, 28, 31, 36
- updateRock(), 17
- updateSelection, 4, 12, 29
- updateSpin, 17, 19, 28, 31, 36
- updateSpin(), 19
- updateStage, 23, 27, 32, 34
- updateStage(), 22
- updateVisibility, 23, 27, 33, 34
- updateZoomMove, 17, 19, 28, 31, 35
- zoomMove, 15, 18, 19, 37
- zoomMove(), 15, 18, 19, 36