

404NotFound

Premi: better than Prezi.



Definizione di Prodotto

Versione	1.0
Redazione	Gobbo Ismaele Rettore Andrea Vegro Federico Manuto Monica De Lazzari Enrico Cossu Mattia Camborata Marco
Verifica	Vegro Federico Camborata Marco
Responsabile	Cossu Mattia
Uso	Esterno
Stato	Formale
Ultima modifica	21 agosto 2015
Lista di distribuzione	404NotFound prof. Tullio Vardanega prof. Riccardo Cardin Zucchetti S.p.a.

Registro delle modifiche

Versione	Autore	Data	Descrizione
1.0	Cossu Mattia	21-08-2015	Approvazione documento
0.1	Gobbo Ismaele	18-06-2015	Stesura scheletro, scrittura introduzione al documento

Tabella 1: Storico versioni del documento.

Indice

1	Introduzione	5
1.1	Scopo del documento	5
1.2	Scopo del Progetto	5
1.3	Glossario	5
1.4	Riferimenti	5
1.4.1	Normativi	5
1.4.2	Informativi	5
2	Standard di Progetto	6
3	Specifica componenti	6
3.1	premi/server	6
3.1.1	premi/server/publish	6
3.2	premi/client	20
3.3	premi/client/presentation	21
3.3.1	premi/client/presentation/lib/databaseAPI	21
3.3.2	premi/client/presentations/lib/OrderedGOList	29
3.4	premi/client/presentationManager	33
3.4.1	premi/client/presentationManager/views/editPresentation.ng . .	33
3.4.2	premi/client/presentationManager/views/newPresentation.ng .	34
3.4.3	premi/client/presentationManager/views/presentationManager.ng	35
3.4.4	premi/client/presentationManager/views/presentations.ng . . .	35
3.4.5	premi/client/presentationManager/views/removePresentation.ng	35
3.4.6	premi/client/presentationManager/controllers/editPresentationCtrl	36
3.4.7	premi/client/presentationManager/controllers/newPresentationCtrl	37
3.4.8	premi/client/presentationManager/controllers/presentationManagerCtrl	38
3.4.9	premi/client/presentationManager/controllers/presentationsCtrl	38
3.4.10	premi/client/presentationManager/controllers/removePresentationCtrl	39
3.5	premi/client/editor	40
3.5.1	premi/client/editor/lib/GObject	40
3.5.2	premi/client/editor/lib/GOProvider	43
3.5.3	premi/client/editor/lib/frame	44
3.5.4	premi/client/editor/lib/GOContainer	49
3.5.5	premi/client/editor/lib/Image	54
3.5.6	premi/client/editor/lib/Infographic	56
3.5.7	premi/client/editor/lib/interactInit	61
3.5.8	premi/client/client/lib/Observer	63
3.5.9	premi/client/editor/lib/saver	64
3.5.10	premi/client/editor/lib/Shape	65
3.5.11	premi/client/editor/lib/Text	68
3.6	premi/client/frameEditor	70
3.7	premi/client/infographicEditor	70
3.8	premi/client/trailsEditor	70
3.9	premi/client/userManager	70
3.10	premi/client/viewer	70



4	Tracciamento	70
5	Diagrammi di Sequenza	70

Elenco delle tabelle

1	Storico versioni del documento.	1
---	---	---

Elenco delle figure

1	Diagramma del package premi/client	6
2	Diagramma della classe premi/server/publish	6
3	Diagramma del package premi/client	20
4	Diagramma del package premi/client/presentation	21
5	Diagramma della classe premi/client/presentation/lib/databaseAPI . .	21
6	Diagramma della classe premi/client/presentations/lib/OrderedGOList	30
7	Diagramma del package premi/client/presentationManager	34
8	Diagramma della classe premi/client/presentationManager/editPresen- tationCtrl	36
9	Diagramma della classe premi/client/presentationManager/newPresen- tationCtrl	37
10	Diagramma della classe premi/client/presentationManager/controller- s/presentationManagerCtrl	38
11	Diagramma della classe premi/client/presentationManager/controller- s/presentationsCtrl	38
12	Diagramma della classe premi/client/presentationManager/removePre- sentationCtrl	39
13	Diagramma della classe premi/client/editor	40
14	Diagramma della classe premi/client/editor/GObject	40
15	Diagramma della classe premi/client/editor/GObject	43
16	Diagramma della classe premi/client/editor/GObject	44
17	Diagramma della classe premi/client/editor/GOContainer	49
18	Diagramma della classe premi/client/editor/GObject	54
19	Diagramma della classe premi/client/editor/GObject	56
20	Diagramma della classe premi/client/editor/GObject	65
21	Diagramma della classe premi/client/editor/GObject	68

1 Introduzione

1.1 Scopo del documento

Questo documento approfondisce la definizione della struttura e dei componenti di Premi già discussa nella *Specifica Tecnica v2.0*. Ogni componente verrà descritto in modo sufficientemente dettagliato da consentire ai programmatori di sviluppare il software in modo coerente con quanto progettato finora.

1.2 Scopo del Progetto

Lo scopo del progetto è la realizzazione di un software di presentazione di slide non basato sul modello di PowerPoint_G, sviluppato in tecnologia HTML5_G e che funzioni sia su desktop che su dispositivo mobile. Il software dovrà permettere la creazione da parte dell'autore e la successiva presentazione del lavoro, fornendo effetti grafici di supporto allo storytelling e alla creazione di mappe mentali.

1.3 Glossario

Al fine di evitare ogni ambiguità relativa al linguaggio e ai termini utilizzati nei documenti formali tutti i termini e gli acronimi presenti nel seguente documento che necessitano di definizione saranno seguiti da una "G" in pedice e saranno riportati in un documento esterno denominato Glossario.pdf. Tale documento accompagna e completa il presente e consiste in un listato ordinato di termini e acronimi con le rispettive definizioni e spiegazioni.

1.4 Riferimenti

1.4.1 Normativi

- **Norme di Progetto:** *NormeDiProgetto_v2.0.pdf*;
- **Capitolato d'appalto C4:** Premi: Software di presentazione "better than Prezi" - <http://www.math.unipd.it/~tullio/IS-1/2014/Progetto/C4.pdf>.

1.4.2 Informativi

- **Slide dell'insegnamento Ingegneria del Software modulo A:**
<http://www.math.unipd.it/~tullio/IS-1/2014/>;
- **Ingegneria del software - Ian Sommerville - 8a Edizione (2007):**
 - Part 4: Software Management.

2 Standard di Progetto

3 Specifica componenti

3.1 premi/server

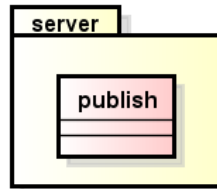


Figura 1: Diagramma del package premi/client

3.1.1 premi/server/publish

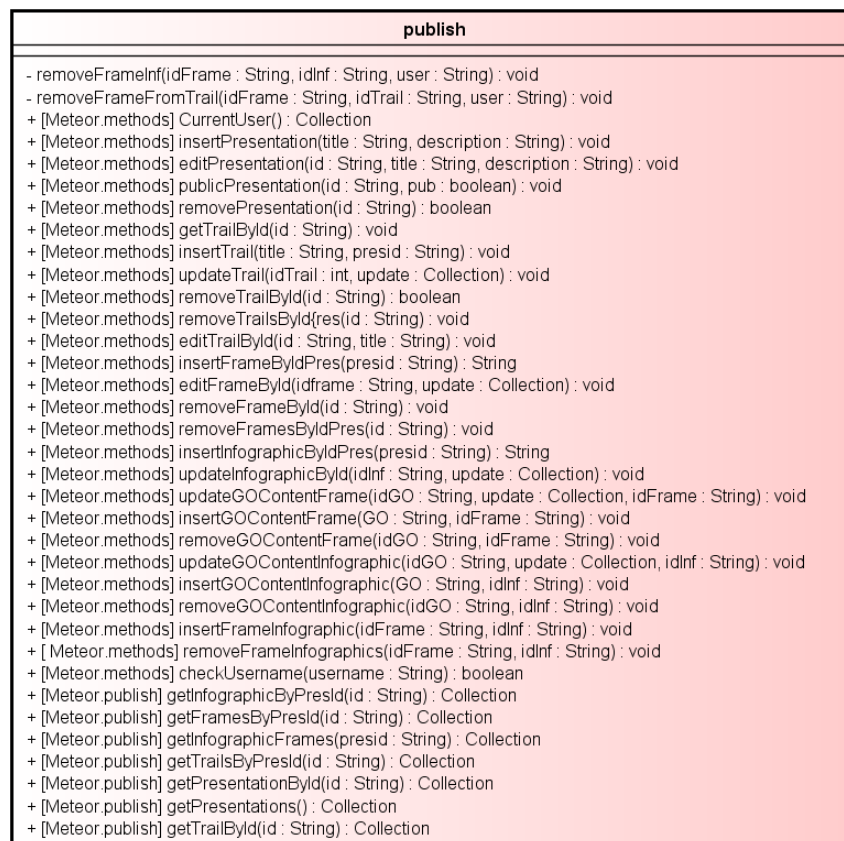


Figura 2: Diagramma della classe premi/server/publish

Descrizione

Lista di metodi che permettono al client di interagire con il database del server. I metodi marcati [**Meteor.methods**] devono essere inseriti nel servizio \$meteor per permettere poi al client di accedervi attraverso il pattern Dependency Injection_G, mentre quelli marcati [**Meteor.publish**] hanno lo scopo di pubblicare al client soltanto le informazioni a cui esso può accedere.

Metodi

- **removeFrameInf(idFrame : String, idInf : String, user : String) : void**

Metodo privato di utilità che rimuove ogni occorrenza di un frame da un'infografica.

Argomenti

- **idFrame : String**
Codice identificativo del frame da rimuovere
- **idInf : String**
Codice identificativo dell'infografica che possiede il frame
- **user : String**
Codice identificativo dell'utente che sta effettuando la rimozione

Note

- Il client non deve poter accedere direttamente a questo metodo

- **removeFrameFromTrail(idFrame : String, idTrail : String, user : String) : void**

Metodo privato di utilità che rimuove ogni occorrenza di un frame da un trail

Argomenti

- **idFrame : String**
Codice identificativo del frame da rimuovere
- **idTrail : String**
Codice identificativo del trail che possiede il frame
- **user : String**
Codice identificativo dell'utente che sta effettuando la rimozione

Note

- Il client non deve poter accedere direttamente a questo metodo

+ **currentUser() : Collection**

Restituisce le informazioni riguardanti l'utente che sta interrogando il database attraverso una collezione di documenti di MongoDB

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando `Meteor.methods`

+ **insertPresentation(title : String, description : String) : String**

Aggiunge una nuova presentazione nel database di proprietà dell'utente, e restituisce il suo codice identificativo

Argomenti

- **title : String**
Titolo della nuova presentazione
- **description : String**
Descrizione della nuova presentazione

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando `Meteor.methods`
- Inizializzare ogni attributo della presentazione:
 - *title*: con il titolo ricevuto
 - *description*: con la descrizione ricevuta
 - *owner*: con l'id dell'utente che sta creando la presentazione
 - *isPublic*: **false**, la presentazione inizialmente è sempre privata
- Ogni presentazione possiede almeno un'infografica, che andrà creata e inizializzata nel database con i seguenti campi dato:
 - *dataX*: -5000
 - *dataY*: -4000
 - *dataZ*: 0
 - *scale*: 0
 - *height*: 7920
 - *width*: 10240
 - *zoom*: 0
 - *presid*: il codice univoco della presentazione
 - *owner*: l'id dell'utente che sta creando la presentazione

- *background*: Collezione(JSON) vuota
- *content*: Collezione(JSON) vuota
- *framesId*: array vuoto
- *type*: infographics

+ **editPresentation(id : String, title : String, description : String) : void**

Modifica le informazioni di una presentazione (titolo e descrizione)

Argomenti

- **title : String**
Nuovo titolo della presentazione
- **description : String**
Nuova descrizione della presentazione

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods

+ **publicPresentation(id : String, pub : boolean) : void**

Rende una presentazione pubblica o privata

Argomenti

- **id : String**
Codice identificativo della presentazione
- **pub : boolean**
Indica se la presentazione è pubblica (**true**) o privata (**false**)

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods

+ **removePresentation(id : String) : boolean**

Rimuove una presentazione dal database. Restituisce un valore che indica se l'operazione è avvenuta con successo

Argomenti

- **id : String**
Codice identificativo della presentazione.

Note

- Devono essere rimossi, assieme alla presentazione, anche tutti gli altri dati ad essa associati (frame, infografica e trails)
- Dev'essere inserito nel servizio **\$meteor** attraverso il comando `Meteor.methods`

+ getTrailById(id : String) : Collection

Restituisce il trail corrispondente al codice identificativo fornito, attraverso una collezione di documenti di MongoDB

Argomenti

- **id : String**
Codice identificativo del trail che si sta cercando

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando `Meteor.methods`

+ insertTrail(title : String, presid : String) : void

Inserisce un nuovo trail associato ad una presentazione all'interno del database

Argomenti

- **title : String**
Titolo del nuovo trail
- **presid : String**
Codice identificativo della presentazione a cui il trail è associato

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando `Meteor.methods`
- inizializzare il trail con i seguenti campi:
 - *title*: con il titolo ricevuto
 - *owner*: con l'id dell'utente che sta chiamando il metodo
 - *presid*: con il codice identificativo della presentazione

– *trail*: è una matrice vuota (`[]`)

+ **updateTrail(idTrail : int, update : Collection) : void**

Aggiorna i dati di un trail con quelli forniti

Argomenti

- **idTrail : String**
Codice identificativo del trail
- **update : Collection**
Insieme dei dati del trail modificati dall'utente

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando `Meteor.methods`

+ **removeTrailById(id : String) : boolean**

Rimuove dal database il trail associato al codice identificativo fornito. Restituisce **true** se l'operazione ha avuto successo, **false** altrimenti.

Argomenti

- **id : String**
Codice identificativo del trail da rimuovere

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando `Meteor.methods`

+ **removeTrailsByIdPres(id : String) : void**

Rimuove dal database ogni trail associato ad una presentazione

Argomenti

- **id : String**
Codice identificativo della presentazione da cui rimuovere ogni trail

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando `Meteor.methods`

+ `editTrailById(id : String, title : String) : void`

Permette all'utente di rinominare un trail in suo possesso

Argomenti

- **id : String**
Codice identificativo del trail da rinominare
- **title : String**
Nuovo titolo, o nome, del trail

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando `Meteor.methods`

+ `insertFrameByIdPres(presid : String) : String`

Inserisce un nuovo frame all'interno di una presentazione, e restituisce il suo id

Argomenti

- **presid : String**
Codice identificativo della presentazione a cui assegnare il nuovo frame

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando `Meteor.methods`
- Inizializzare il frame con i seguenti campi dato:
 - *presid*: con il codice identificativo della presentazione
 - *owner*: con l'id dell'utente che ha effettuato la chiamata al metodo
 - *dataX*: 0
 - *dataY*: 0
 - *dataZ*: 0
 - *height*: 792
 - *width*: 1024
 - *scale*: 1
 - *backgroundColor*: #FFFFFF
 - *content*: Collezione(JSON) vuota
 - *type*: frame
 - *lvl*: 0

+ `editFrameById(idframe : String, update : Collection) : void`

Modifica un frame con nuovi dati inseriti dall'utente

Argomenti

- **idframe : String**
Codice identificativo del frame
- **update : Collection**
Collezione di dati modificati del frame da salvare sul database

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando `Meteor.methods`

+ `removeFrameById(id : String) : void`

Rimuove dal database il frame corrispondente al codice identificativo inviato

Argomenti

- **id : String**
Codice identificativo del frame da rimuovere

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando `Meteor.methods`
- Il frame dev'essere rimosso anche dai trail e dalle infografiche in cui è stato utilizzato. Servirsi dei metodi `removeFrameFromTrail` e `RemoveFramInf`

+ `removeFramesByIdPres(id : String) : void`

Rimuove dal database tutti i frame associati ad una presentazione.

Argomenti

- **id : String**
Codice identificativo della presentazione

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods

+ insertInfographicByIdPres(presid : String) : String

Inserisce un'infografica nel database associandola ad una presentazione, e restituisce il suo codice identificativo

Argomenti

- **presid : String**
Codice identificativo della presentazione

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods
- L'Infografica dev'essere inizializzata con i seguenti campi dato:
 - *presid*: il codice identificativo della presentazione
 - *owner*: Il codice identificativo dell'utente che ha chiamato il metodo
 - *content*: Collezione(JSON) vuota
 - *frames*: Collezione(JSON) vuota
 - *type*: infographic

+ updateInfographicById(idInf : String, update : Collection) : void

Aggiorna un'infografica con campi dati modificati dall'utente

Argomenti

- **idInf : String**
Codice identificativo dell'infografica da aggiornare
- **update : Collection**
Collezione di dati con cui aggiornare l'infografica

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods

+ updateGOContentFrame(idGO : String, update : Collection, idFrame : String) : void

Aggiorna un oggetto grafico contenuto all'interno di una presentazione con campi dati modificati dall'utente

Argomenti

- **idGO : String**
Codice identificativo dell'oggetto grafico da modificare
- **update : Collection**
Collezione di dati modificati dell'oggetto grafico
- **idFrame : String**
Codice identificativo del Frame che contiene l'oggetto grafico

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods

+ **insertGOContentFrame(GO : Collection, idFrame : String) : void**

Inserisce un oggetto grafico all'interno di un frame

Argomenti

- **GO : Collection**
Oggetto grafico convertito in JSON
- **idFrame : String**
Codice identificativo del frame in cui inserire l'oggetto grafico

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods
- Ogni oggetto grafico possiede dei metodi per la conversione in JSON. Quest'operazione va sempre effettuata dal client

+ **removeGOContentFrame(idGO : String, idFrame : String) : void**

Rimuove un oggetto grafico dal frame che lo contiene

Argomenti

- **idGO : String**
Codice identificativo dell'oggetto grafico
- **idFrame : String**
Codice identificativo del frame che contiene l'oggetto grafico

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods

+ **updateGOContentInfographic(idGO : String, update : Collection, idInf : String) : void**

Aggiorna un oggetto grafico contenuto all'interno di un'infografica

Argomenti

- **idGO : String**
Codice identificativo dell'oggetto grafico
- **update : Collection**
Collezione di dati dell'oggetto grafico modificati dall'utente
- **idInf : String**
Codice identificativo dell'infografica

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods

+ **insertGOContentInfographic(GO : Collection, idInf : String) : void**

Inserisce un oggetto grafico all'interno di un'infografica

Argomenti

- **GO : Collection**
Oggetto grafico convertito in JSON
- **idInf : String**
Codice identificativo dell'infografica

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods
- Ogni oggetto grafico possiede dei metodi per la conversione in JSON. Quest'operazione va sempre effettuata dal client

+ **removeGOContentInfographic(idGO : String, idInf : String) : void**

Rimuove un oggetto grafico da un'infografica

Argomenti

- **idGO : String**
Codice identificativo dell'oggetto grafico
- **idInf : String**
Codice identificativo dell'infografica

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods

+ insertFrameInfographic(idFrame : String, idInf : String) : void

Inserisce un frame all'interno di un'infografica

Argomenti

- **idFrame : String**
Codice identificativo del frame
- **idInf : String**
Codice identificativo dell'infografica

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods

+ removeFrameInfographics(idFrame : String, idInf : String) : void

Rimuove un frame da un'infografica

Argomenti

- **idFrame : String**
Codice identificativo del frame
- **idInf : String**
Codice identificativo dell'infografica

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods

+ checkUsername(username : String) : boolean

Verifica che l'username ricevuto sia presente tra quelli registrati nel database.

Argomenti

- **username : String**

Il codice identificativo dell'username da verificare

Note

- Dev'essere inserito nel servizio **\$meteor** attraverso il comando Meteor.methods

+ **getInfographicByPresId(id : String) : Collection**

Pubblica una lista di infografiche associate ad una presentazione

Argomenti

- **id : String**

Il codice identificativo della presentazione associata alle infografiche

Note

- Dev'essere inserito come funzione **publishfunction** in Meteor.publish

+ **getFramesByPresId(id : String) : Collection**

Pubblica una lista di frames associati ad una presentazione

Argomenti

- **id : String**

Il codice identificativo della presentazione associata ai frame

Note

- Dev'essere inserito come funzione **publishfunction** in Meteor.publish
- La presentazione deve appartenere all'utente che ha effettuato la richiesta

+ **getInfographicFrames(presid : String) : Collection**

Pubblica una lista di tutti i frames delle infografiche di una presentazione

Argomenti

- **presid : String**

Il codice identificativo della presentazione associata alle infografiche

Note

- Dev'essere inserito come funzione **publishfunction** in Meteor.publish
- La presentazione deve appartenere all'utente che ne ha effettuato la richiesta

+ **getTrailsByPresId(id : String) : Collection**

Pubblica una lista di trails associati ad una presentazione

Argomenti

- **id : String**

Il codice identificativo della presentazione associata ai trails

Note

- Dev'essere inserito come funzione **publishfunction** in Meteor.publish
- La presentazione deve appartenere all'utente che ne ha effettuato la richiesta

+ **getPresentationsById(id : String) : Collection**

Pubblica una presentazione all'utente

Argomenti

- **id : String**

Il codice identificativo della presentazione

Note

- Dev'essere inserito come funzione **publishfunction** in Meteor.publish
- La presentazione deve appartenere all'utente che ne ha effettuato la richiesta

+ **getPresentations() : Collection**

Pubblica una lista di tutte le presentazioni che l'utente possiede

Note

- Dev'essere inserito come funzione **publishfunction** in Meteor.publish
- Le presentazioni devono appartenere all'utente che ne ha effettuato la richiesta

+ getTrailsById(id : String) : Collection

Pubblica un trail all'utente che ne ha effettuato la richiesta

Argomenti

- **id : String**
Il codice identificativo del trail da pubblicare

Note

- Dev'essere inserito come funzione **publishfunction** in Meteor.publish
- Il trail deve appartenere all'utente che ne ha effettuato la richiesta

3.2 premi/client

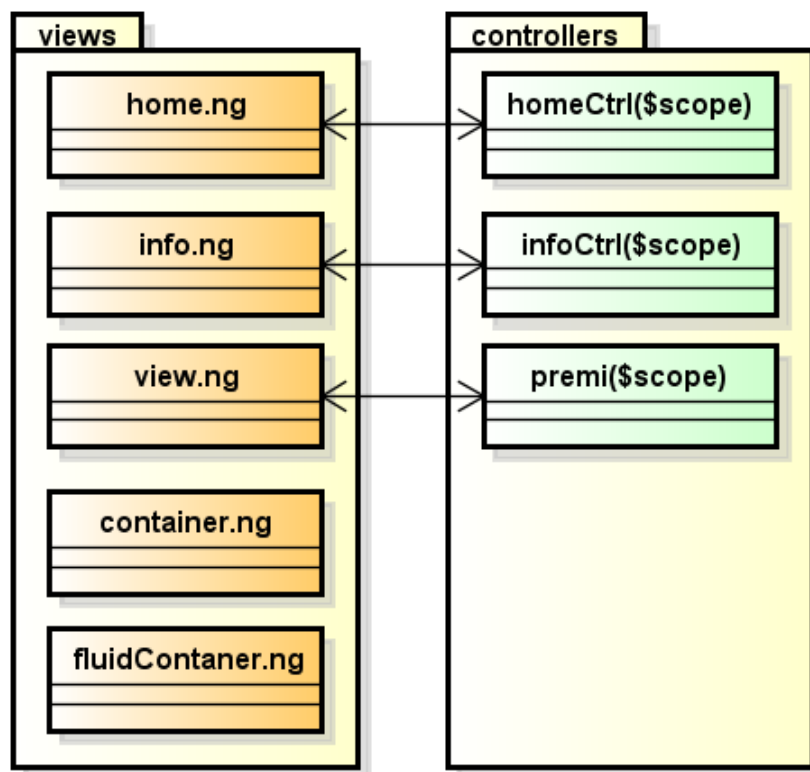


Figura 3: Diagramma del package premi/client

package ancora incompleto
package ancora incompleto

```

package ancora incompleto
package ancora incompleto
package ancora incompleto
package ancora incompleto
package ancora incompleto

```

3.3 premi/client/presentation

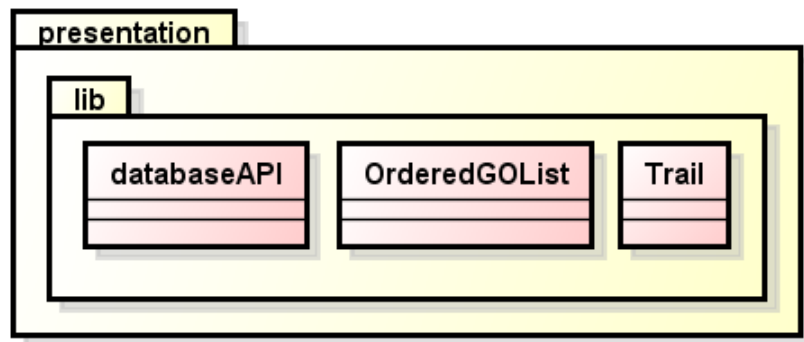


Figura 4: Diagramma del package premi/client/presentation

3.3.1 premi/client/presentation/lib/databaseAPI

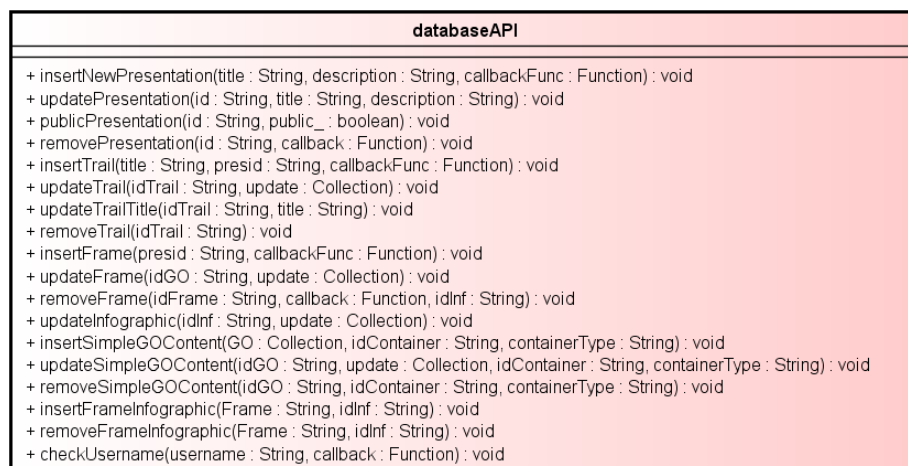


Figura 5: Diagramma della classe premi/client/presentation/lib/databaseAPI

Descrizione

Classe di metodi statici che permette al client di interfacciarsi ai metodi corrispondenti del server

Metodi

+ **insertNewPresentation(title : String, description : String, callbackFunc : Function) : void**

Permette l'inserimento di una nuova presentazione all'interno del database

Argomenti

- **title : String**
Titolo della nuova presentazione
- **description : String**
Descrizione della nuova presentazione
- **callbackFunc : Function**
Funzione di callback_G per la restituzione dell'id della presentazione

Note

- Chiama il metodo *insertPresentation* pubblicato in *\$meteor*

+ **updatePresentation(id : String, title : String, description : String) : void**

Permette l'aggiornamento del titolo e della descrizione di una presentazione dell'utente

Argomenti

- **id : String**
Codice identificativo della presentazione da aggiornare
- **title : String**
Nuovo titolo della presentazione
- **description : String**
Nuova descrizione della presentazione

Note

- Chiama il metodo *editPresentation* pubblicato in *\$meteor*

+ **publicPresentation(id : String, public_ : boolean) : void**

Rende una presentazione pubblica o privata

Argomenti

- **id : String**
Codice identificativo della presentazione da aggiornare
- **public_ : boolean**
Variabile booleana: se *true* la presentazione verrà resa pubblica, se *false* verrà resa privata

Note

- Chiama il metodo *publicPresentation* pubblicato in *\$meteor*

+ **removePresentation(id : String, callback : Function) : void**

Permette la rimozione di una presentazione dell'utente dal database

Argomenti

- **id : String**
Codice identificativo della presentazione da rimuovere
- **callbackFunc : Function**
Funzione di callback_G per la restituzione di una conferma dell'avvenuta rimozione della presentazione

Note

- Chiama il metodo *removePresentation* pubblicato in *\$meteor*

+ **insertTrail(title : String, presid : String, callbackFunc : Function) : void**

Permette l'inserimento di un trail all'interno del database

Argomenti

- **title : String**
Titolo del trail da inserire
- **presid : String**
Codice identificativo della presentazione associata al trail
- **callbackFunc : Function**
Funzione di callback_G per la restituzione del codice identificativo del nuovo trail creato

Note

- Chiama il metodo *insertTrail* pubblicato in *\$meteor*

+ **updateTrail(idTrail : String, update : Collection) : void**

Permette l'aggiornamento di un trail

Argomenti

- **idTrail : String**
Codice identificativo del trail da aggiornare
- **update : Collection**
Collezione di MongoDB degli attributi aggiornati del trail

Note

- Chiama il metodo *updateTrail* pubblicato in *\$meteor*

+ **updateTrailTitle(idTrail : String, title : String) : void**

Permette la modifica del titolo di un trail

Argomenti

- **idTrail : String**
Codice identificativo del trail da aggiornare
- **title : String**
Nuovo titolo del trail

Note

- Chiama il metodo *updateTrailById* pubblicato in *\$meteor*

+ **removeTrail(idTrail : String) : void**

Permette la rimozione di un trail dal database

Argomenti

- **idTrail : String**
Codice identificativo del trail da rimuovere

Note

- Chiama il metodo *removeTrailById* pubblicato in *\$meteor*

+ insertFrame(presid : String, callbackFunc : Function) : void

Permette l'inserimento di un frame all'interno del database

Argomenti

- **presid : String**
Codice identificativo della presentazione associata al nuovo frame
- **callbackFunc : Function**
Funzione di callback_G per la restituzione del codice identificativo del nuovo frame

Note

- Chiama il metodo *insertFrameByIdPres* pubblicato in *\$meteor*

+ updateFrame(idGO : String, update : Collection) : void

Permette l'aggiornamento di un frame all'interno del database

Argomenti

- **idGO : String**
Codice identificativo del frame da aggiornare
- **update : Collection**
Collezione di MongoDB di attributi aggiornati

Note

- Chiama il metodo *editFrameById* pubblicato in *\$meteor*

+ removeFrame(idFrame : String, callback : Function, idInf : String) : void

Permette la rimozione di un frame dal database

Argomenti

- **idFrame : String**
Codice identificativo del frame da rimuovere
- **callback : Function**
Funzione di callback_G per la restituzione del codice identificativo del frame

Note

- Chiama il metodo *removeFrameInfographic* pubblicato in *\$meteor* per rimuovere le associazioni del frame con l'infografica della presentazione
- Chiama il metodo *removeFrameById* pubblicato in *\$meteor* per rimuovere il frame dal database

+ **updateInfographic(idInf : String, update : Collection) : void**

Permette l'aggiornamento di un'infografica

Argomenti

- **idInf : String**
Codice identificativo dell'infografica da aggiornare
- **update : Collection**
Collezione di MongoDB di attributi aggiornati

Note

- Chiama il metodo *updateInfographicById* pubblicato in *\$meteor*

+ **insertSimpleGOContent(GO : Collection, idContainer : String, containerType : String) : void**

Inserisce un oggetto grafico all'interno del database

Argomenti

- **GO : Collection**
Collezione degli attributi dell'oggetto grafico
- **idContainer : String**
Codice identificativo del contenitore in cui inserire l'oggetto grafico
- **ContainerType : String**
Tipo del contenitore: può essere un frame(*frame*) o un'infografica (*infographic*)

Note

- Chiama il metodo *insertGOContentFrame* pubblicato in *\$meteor* se il contenitore è un frame, mentre chiama il metodo *insertGOContentInfographic* se il contenitore è un'infografica

+ **updateSimpleGOContent(idGO : String, update : Collection, idContainer : String, containerType : String) : void**

Aggiorna un oggetto grafico con gli attributi modificati dall'utente

Argomenti

- **idGO : String**
Codice identificativo dell'oggetto grafico da aggiornare
- **update : Collection**
Collezione di MongoDB di attributi aggiornati
- **idContainer : String**
Codice identificativo del contenitore in cui l'oggetto grafico è stato inserito
- **ContainerType : String**
Tipo del contenitore: può essere un frame(*frame*) o un'infografica (*infographic*)

Note

- Chiama il metodo *updateGOContentFrame* pubblicato in *\$meteor* se il contenitore è un frame, mentre chiama il metodo *updateGOContentInfographic* se il contenitore è un'infografica

+ **removeSimpleGOContent(idGO : String, update : Collection, idContainer : String, containerType : String) : void**

Aggiorna un oggetto grafico con gli attributi modificati dall'utente

Argomenti

- **idGO : String**
Codice identificativo dell'oggetto grafico da aggiornare
- **update : Collection**
Collezione di MongoDB di attributi aggiornati
- **idContainer : String**
Codice identificativo del contenitore in cui l'oggetto grafico è stato inserito
- **ContainerType : String**
Tipo del contenitore: può essere un frame(*frame*) o un'infografica (*infographic*)

Note

- Chiama il metodo *updateGOContentFrame* pubblicato in *\$meteor* se il contenitore è un frame, mentre chiama il metodo *updateGOContentInfographic* se il contenitore è un'infografica

+ **removeSimpleGOContent(idGO : String, idContainer : String, containerType : String) : void**

Rimuove un oggetto grafico da un frame o da un'infografica

Argomenti

- **idGO : String**
Codice identificativo dell'oggetto grafico da rimuovere
- **idContainer : String**
Codice identificativo del contenitore in cui l'oggetto grafico è stato inserito
- **ContainerType : String**
Tipo del contenitore: può essere un frame(*frame*) o un'infografica (*infographic*)

Note

- Chiama il metodo *removeGOContentFrame* pubblicato in *\$meteor* se il contenitore è un frame, mentre chiama il metodo *removeGOContentInfographic* se il contenitore è un'infografica

+ insertFrameInfographic(Frame : String, idInf : String) : void

Inserisce un frame all'interno di un'infografica

Argomenti

- **Frame : String**
Codice identificativo del frame
- **idInf : String**
Codice identificativo dell'infografica

Note

- Chiama il metodo *insertFrameInfographic* pubblicato in *\$meteor*

+ removeFrameInfographic(Frame : String, idInf : String) : void

Rimuove un frame all'interno di un'infografica

Argomenti

- **Frame : String**
Codice identificativo del frame
- **idInf : String**
Codice identificativo dell'infografica

Note

- Chiama il metodo *removeFrameInfographic* pubblicato in *\$meteor*

+ **checkUsername(username : String, callback : Function) : void**

Verifica la presenza di uno username nel database

Argomenti

- **username : String**
Nome utente da cercare nel database
- **callback : Function**
Funzione *callback_G* per confermare la presenza o l'assenza dello username nel database

Note

- Chiama il metodo *checkUsername* pubblicato in *\$meteor*. Restituisce una variabile booleana che andrà passata alla funzione *callback_G*

3.3.2 premi/client/presentations/lib/OrderedGOList

Descrizione

Questa classe gestisce una lista ordinata di oggetti grafici da poter essere utilizzata per i frame e l'infografica di una presentazione.

Dipendenze

- **premi/client/editor/lib/Observer**: per l'invio di segnali che avvertono gli altri componenti delle modifiche apportate agli oggetti grafici

Attributi

- **GOarray : Array**

Array di oggetti grafici che compongono la lista ordinata

- **orderBy : String**

Indica il nome dell'attributo che è stato scelto per ordinare gli oggetti grafici

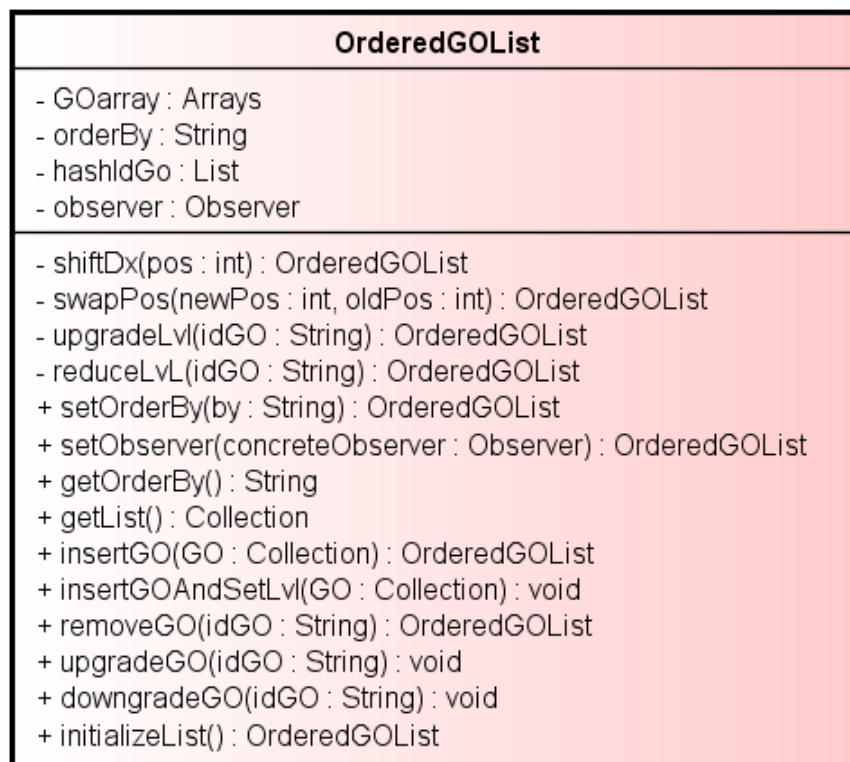


Figura 6: Diagramma della classe premi/client/presentations/lib/OrderedGOList

- hashIdGo : List

Oggetto Hash_G Javascript_G che contiene una lista degli id degli oggetti grafici presenti nell'array, associati alla loro posizione all'interno dell'array

- observer : Observer

Contiene una lista di segnali, ognuno dei quali è associato ad uno specifico oggetto grafico.

Metodi

- shiftDx(pos : int) : OrderedGOList

Sposta l'oggetto grafico presente nella posizione ricevuta nella posizione successiva dell'array. Restituisce un riferimento al *this* per chiamate multiple ai metodi della classe

Argomenti

- **pos : int**
Posizione dell'oggetto da spostare

- swapPos(newPos : int, oldPos : int) : OrderedGOList

Scambia di posizione un oggetto grafico. Restituisce un riferimento al *this* per chiamate multiple ai metodi della classe

Argomenti

- **newPos : int**
La nuova posizione in cui spostare l'oggetto grafico
- **oldPos : int**
La posizione in cui si trova l'oggetto grafico prima dell'esecuzione del metodo

- upgradeLvl(idGO : String) : OrderedGOList

Incrementa il livello di visibilità dell'oggetto grafico associato al codice identificativo ricevuto. Restituisce un riferimento al *this* per chiamate multiple ai metodi della classe

Argomenti

- **idGO : String**
Il codice identificativo dell'oggetto grafico da modificare

- reduceLvl(idGO : String) : OrderedGOList

Riduce il livello di visibilità dell'oggetto grafico associato al codice identificativo ricevuto. Restituisce un riferimento al *this* per chiamate multiple ai metodi della classe

Argomenti

- **idGO : String**
Il codice identificativo dell'oggetto grafico da modificare

+ setOrderBy(by : String) : OrderedGOList

Cambia l'attributo con il quale la classe ordina gli oggetti grafici. Restituisce un riferimento al *this* per chiamate multiple ai metodi della classe

Argomenti

- **by : String**
Il nome dell'attributo con il quale si intende stabilire l'ordine degli oggetti grafici

+ addObserver(concreteObserver : Observer) : OrderedGOList

Imposta l'Observer della classe. Restituisce un riferimento al *this* per chiamate multiple ai metodi della classe

Argomenti

- **concreteObserver : Observer**

L'oggetto Observer da associare alla classe

+ getOrderBy() : String

Restituisce il nome dell'attributo col quale si sta effettuando l'ordinamento degli oggetti grafici

+ getList() : Collection

Restituisce l'array degli oggetti grafici inseriti finora all'interno della lista.

+ insertGO(GO : Collection) : OrderedGOList

Inserisce un oggetto grafico convertito precedentemente in formato $JSON_G$ all'interno della lista. Restituisce un riferimento al *this* per chiamate multiple ai metodi della classe

Argomenti

- **GO : Collection**

L'oggetto grafico da inserire nella lista, già convertito in formato JSON

+ insertGOAndSetLvl(GO : Collection) : void

Inserisce un oggetto grafico convertito precedentemente in formato $JSON_G$ alla fine della lista, e tramite l'Observer invia un segnale di cambio livello dell'oggetto

Argomenti

- **GO : Collection**

L'oggetto grafico da inserire nella lista, già convertito in formato JSON

+ removeGO(idGO : String) : OrderedGOList

Rimuove un oggetto grafico dalla lista. Restituisce un riferimento al *this* per chiamate multiple ai metodi della classe

Argomenti

- **idGO : String**

Il codice identificativo dell'oggetto grafico da rimuovere dalla lista

+ **upgradeGO(idGO : String) : void**

Incrementa di posizione un oggetto grafico nella lista

Argomenti

- **idGO : String**

Il codice identificativo dell'oggetto grafico da incrementare di posizione

Note

- Sfrutta il metodo privato *swapPos* per lo spostamento dell'oggetto grafico

+ **downgradeGO(idGO : String) : void**

Decrementa di posizione un oggetto grafico nella lista

Argomenti

- **idGO : String**

Il codice identificativo dell'oggetto grafico da decrementare di posizione

Note

- Sfrutta il metodo privato *swapPos* per lo spostamento dell'oggetto grafico

+ **initializeList() : OrderedGOList**

Inizializza la lista svuotando *GOarray* e *hashIdGO*. Restituisce un riferimento al *this* per chiamate multiple ai metodi della classe

3.4 premi/client/presentationManager

3.4.1 premi/client/presentationManager/views/editPresentation.ng

Descrizione

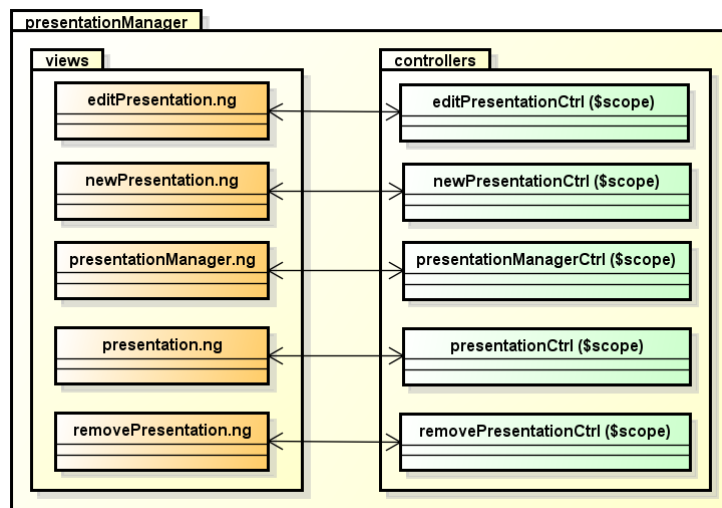


Figura 7: Diagramma del package premi/client/presentationManager

Template della vista associata allo *\$scope* di *editPresentationCtrl*. Permette all'utente di modificare titolo e descrizione di una presentazione.

Note

- Mostra il titolo della presentazione in un input $HTML_G$, modificabile, attraverso l'attributo dello *\$scope* Presentation.title
- Mostra la descrizione della presentazione in un input $HTML_G$, modificabile, attraverso l'attributo dello scope Presentation.description
- Possiede un bottone associato al metodo *save()* dello *\$scope* per salvare la presentazione
- Possiede un bottone associato al metodo *discard()* dello *\$scope* per annullare le modifiche effettuate sulla presentazione

3.4.2 premi/client/presentationManager/views/newPresentation.ng

Descrizione

Template della vista associata allo *\$scope* di *newPresentationCtrl*. Permette all'utente di creare una nuova presentazione, fornendo un titolo e una descrizione.

Note

- Mostra un input HTML nel quale inserire il titolo della presentazione, che va associato all'attributo *title* dello *\$scope*
- Mostra un input HTML nel quale inserire la descrizione della presentazione, che va associata all'attributo *description* dello *\$scope*

- Possiede un bottone associato al metodo *save()* dello *\$scope* per salvare la presentazione
- Possiede un bottone associato al metodo *discard()* dello *\$scope* per annullare la creazione della presentazione

3.4.3 premi/client/presentationManager/views/presentationManager.ng

Descrizione

Template della vista associata allo *\$scope* di *presentationmagerCtrl*. Fornisce uno scheletro per le altre viste dedicate alla gestione delle presentazioni

3.4.4 premi/client/presentationManager/views/presentations.ng

Descrizione

Template della vista associata allo *\$scope* di *presentationsCtrl*. Mostra una lista di tutte le presentazioni dell'utente

Note

- Mostra la lista delle presentazioni, le quali sono contenute nell'attributo *Presentations* dello *\$scope*
- Per ogni presentazione fornisce dei link per sua la modifica e rimozione

3.4.5 premi/client/presentationManager/views/removePresentation.ng

Descrizione

Template della vista associata allo *\$scope* di *removePresentationCtrl*. Permette all'utente di rimuovere una presentazione dal database.

Note

- Mostra un messaggio di conferma prima di rimuovere la presentazione
- Dopo il messaggio di conferma mostra un tasto associato al metodo *remove()* dello *\$scope* per la rimozione definitiva della presentazione
- Dopo il messaggio di conferma mostra un tasto associato al metodo *discard()* dello *\$scope* per annullare il processo di rimozione della presentazione

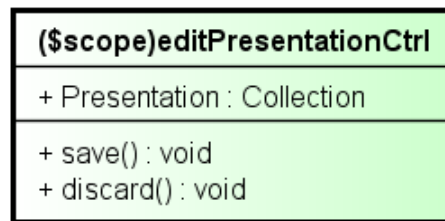


Figura 8: Diagramma della classe premi/client/presentationManager/editPresentationCtrl

3.4.6 premi/client/presentationManager/controllers/editPresentationCtrl

Descrizione

Controller della view *editPresentation.ng*. Permette all'utente di modificare titolo e descrizione di una presentazione.

Associazioni

- **client/presentation/lib/databaseAPI**: per salvare la presentazione modificata

Attributi

+ Presentation : Collection

Presentazione che l'utente intende modificare. Viene inizializzata dal controller tramite il codice identificativo passato come parametro dal browser (servizio *\$stateParams* di AngularJS)

Metodi

+ save() : void

Utilizza il metodo *+ updatePresentation(id, title, description)* di *databaseAPI* per il salvataggio della presentazione nel database. Rimanda poi alla lista delle presentazioni utilizzando l'oggetto *\$state* di *\$stateProvider*

+ discard() : void

Annulla le modifiche effettuate dall'utente sul titolo e sulla descrizione della presentazione. Rimanda poi alla lista delle presentazioni utilizzando l'oggetto *\$state* di *\$stateProvider*

3.4.7 premi/client/presentationManager/controllers/newPresentationCtrl

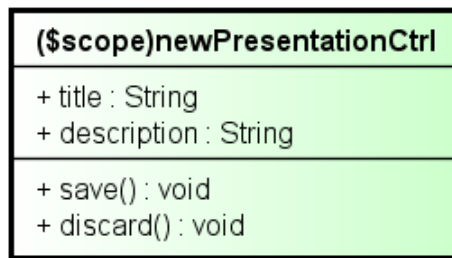


Figura 9: Diagramma della classe premi/client/presentationManager/newPresentationCtrl

Descrizione

Controller della view *editPresentation.ng*. Permette all'utente di creare una nuova presentazione e di salvarla nel database.

Associazioni

- **client/presentation/lib/databaseAPI**: per salvare la presentazione nel database

Attributi

+ **title** : **String**

Il titolo della nuova presentazione

+ **description** : **String**

La descrizione della nuova presentazione

Metodi

+ **save()** : **void**

Utilizza il metodo *+ insertNewPresentation(title, description)* di databaseAPI per il salvataggio della presentazione nel database. Rimanda poi alla lista delle presentazioni utilizzando l'oggetto *\$state* di *\$stateProvider*

+ discard() : void

Annulla le modifiche effettuate dall'utente sul titolo e sulla descrizione della presentazione. Rimanda poi alla lista delle presentazioni utilizzando l'oggetto *\$state* di *\$stateProvider*

3.4.8 premi/client/presentationManager/controllers/presentationManagerCtrl

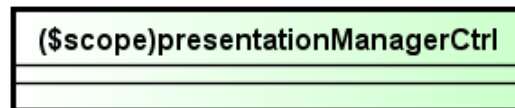


Figura 10: Diagramma della classe premi/client/presentationManager/controllers/-presentationManagerCtrl

Descrizione

Questo controller non è al momento provvisto di funzionalità. Si appoggia alla vista associata *presentationManager.ng*, la quale funge da scheletro per le viste necessarie alla gestione delle presentazioni dell'utente.

3.4.9 premi/client/presentationManager/controllers/presentationsCtrl

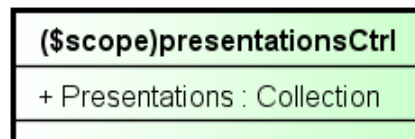


Figura 11: Diagramma della classe premi/client/presentationManager/controllers/-presentationsCtrl

Descrizione

Fornisce alla vista associata *presentations.ng* una lista di tutte le presentazioni in possesso dell'utente

Attributi

+ Presentations : Collection

Collezione MongoDB di presentazioni. Viene inizializzata dal controller prelevando le presentazioni pubblicate al client.

3.4.10 premi/client/presentationManager/controllers/removePresentationCtrl

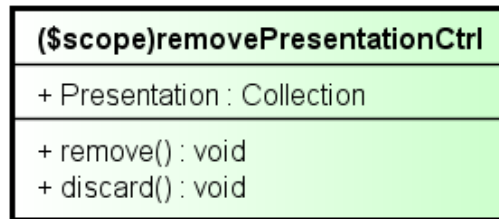


Figura 12: Diagramma della classe premi/client/presentationManager/removePresentationCtrl

Descrizione

Controller della view *removePresentation.ng*. Permette all'utente di eliminare una presentazione da lui creata in precedenza.

Associazioni

- **client/presentation/lib/databaseAPI**: per effettuare la rimozione sul database

Attributi

+ Presentation : Collection

Presentazione che l'utente intende rimuovere. Viene inizializzata dal controller tramite il codice identificativo passato come parametro dal browser (servizio *\$stateParams* di AngularJS)

Metodi

+ remove() : void

Utilizza il metodo *+ removePresentation(id, title, description)* di *databaseAPI* per la rimozione della presentazione dal database. Rimanda poi alla lista delle presentazioni utilizzando l'oggetto *\$state* di *\$stateProvider*

+ discard() : void

Annulla le modifiche effettuate dall'utente sulla presentazione. Rimanda poi alla lista delle presentazioni utilizzando l'oggetto *\$state* di *\$stateProvider*

3.5 premi/client/editor

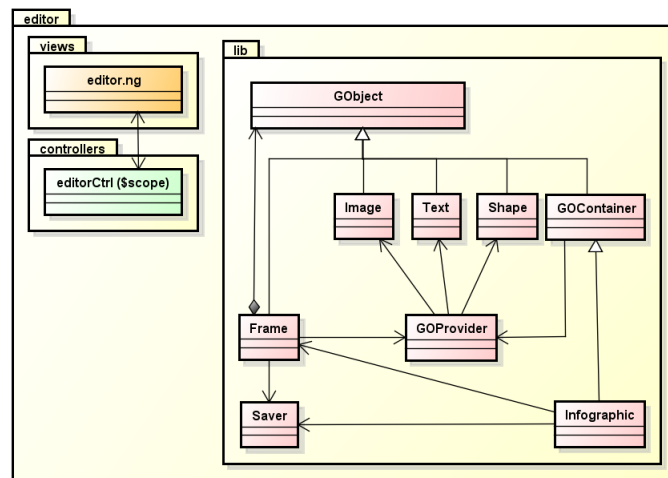


Figura 13: Diagramma della classe premi/client/editor

3.5.1 premi/client/editor/lib/GObject

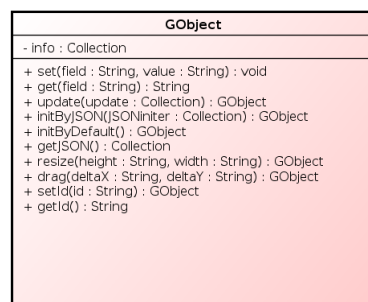


Figura 14: Diagramma della classe premi/client/editor/GObject

Descrizione

GObject è una classe astratta che rappresenta un oggetto generico della presentazione. Contiene i metodi generali che caratterizzano ciascun oggetto grafico che può essere inserito in una presentazione.

Attributi

- info : Collection

info è un oggetto JSON che contiene i seguenti campi:

- *_id*: rappresenta l'id che identifica l'oggetto;
- *dataX*: identifica la posizione orizzontale dell'asse x dell'oggetto;
- *dataY*: identifica la posizione verticale dell'asse y dell'oggetto;
- *dataZ*: identifica il grado di trasparenza dell'oggetto;
- *height*: identifica l'altezza dell'oggetto;
- *width*: identifica la larghezza dell'oggetto;
- *scale*: identifica la scala dell'oggetto;
- *lvl*: identifica il livello dell'oggetto.

Metodi

+ **set(field : String, value : String) : void**

permette di settare un campo dell'attributo info.

Argomenti

- **field : String**
field identifica il campo da settare di info;
- **value : String**
value rappresenta il valore del campo da settare su l'attributo info.

-

+ **get(field : String) : String**

restituisce il valore di un campo dell'attributo info.

Argomenti

- **field : String**
field identifica l'attributo info di cui si vuole venga restituito il valore.

+ **update(update : Collection) : GObject**

permette di aggiornare i campi dell'attributo info.

Argomenti

- **update : Collection**
update è un oggetto JSON che contiene chiave e valore dei campi che devono essere aggiornati.

+ **initByJSON(JSONNiter : Collection) : GObject**

permette di inizializzare l'attributo info tramite un oggetto JSON.

Argomenti

- **JSONNiter : Collection**

è un oggetto JSON che contiene chiave e valore di inizializzazione dei campi dell'attributo info.

+ **initByDefault() : GObject**

permette di inizializzare i campi dell'attributo info con i parametri di default.

+ **getJSON() : String**

restituisce la collection dell'attributo info.

+ **resize(height : String, width : String) : GObject**

permette di settare l'altezza e la larghezza dell'oggetto.

Argomenti

- **height : String**

identifica il valore dell'altezza da settare sul campo height dell'attributo info;

- **width : String**

identifica il valore dell'altezza da settare sul campo width dell'attributo info;

+ **drag(deltaX : String, deltaY : String) : GObject**

permette di settare la posizione dell'oggetto sull'asse x e y.

Argomenti

- **deltaX : String**

identifica il valore della posizione sull'asse x da settare sul campo deltaX dell'attributo info.

- **deltaY : String**

identifica il valore della posizione sull'asse y da settare sul campo deltaY dell'attributo info;

+ setId(id : String) : GObject

permette di modificare o di settare l'id dell'oggetto

Argomenti

- **id : String**
identifica il valore dell'id da settare sul campo _id dell'attributo info.

+ getId()

restituisce l'id dell'oggetto.

3.5.2 premi/client/editor/lib/GOProvider

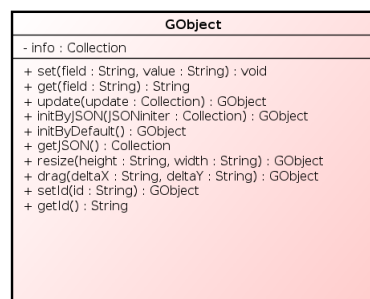


Figura 15: Diagramma della classe premi/client/editor/GObject

Descrizione

GOProvider è una classe statica contiene i metodi per inizializzare gli oggetti image, text, shape che possono essere inseriti in un frame.

Metodi

+ init(type : String) : GObject

inizializza con i parametri di default un oggetto image, shape o text e ne restituisce il riferimento.

Argomenti

- **type : String**
identifica il tipo di oggetto da inizializzare.

+ initByJSON(GO : Collection) : GObject

inizializza con un oggetto JSON, un oggetto image, text o shape e ne restituisce il riferimento.

Argomenti

- **GO : Collection**

è un oggetto JSON che serve per inizializzare l'oggetto da restituire come riferimento.

3.5.3 premi/client/editor/lib/frame

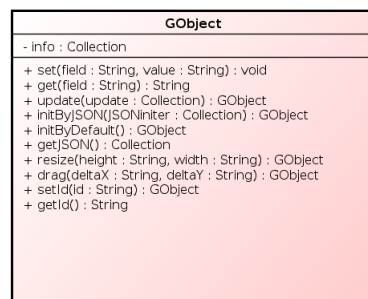


Figura 16: Diagramma della classe premi/client/editor/GObject

Descrizione

frame è una classe che rappresenta un frame di una presentazione. E' un oggetto che può essere rappresentato nella presentazione.

Classi ereditate

- GObject

Attributi

- info : Collection

l'attributo info è un oggetto JSON che estende l'attributo info ereditato da GObject. I campi aggiuntivi sono:

- *backgroundColor*: rappresenta il colore di Background del frame;
- *content*: è un oggetto JSON che contiene gli oggetti che fanno parte del frame;

- *type*: identifica che l'oggetto trattato è un frame.

- **selectedGO : GObject**

contiene l'oggetto GObject contenuto nel frame corrente selezionato dall'utente.

- **saver : Saver**

contiene un oggetto saver che permette di interfacciarsi con il database.

Metodi

- **findNewId() : String**

trova un nuovo id valido per il frame corrente.

+ **set(field : String, value : String) : void**

permette di settare un campo dell'attributo info.

Argomenti

- **field : String**
field identifica il campo da settare di info;
- **value : String**
value rappresenta il valore del campo da settare su l'attributo info.

+ **get(field : String) : String**

restituisce il valore di un campo dell'attributo info.

Argomenti

- **field : String**
field identifica l'attributo info di cui si vuole venga restituito il valore.

+ **update(update : Collection) : frame**

permette di aggiornare i campi dell'attributo info.

Argomenti

- **update : Collection**

update è un oggetto JSON che contiene chiave e valore dei campi che devono essere aggiornati.

+ **initByJSON(frameJSON : Collection, disableSaver: boolean) : frame**

permette di inizializzare l'attributo info tramite un oggetto JSON.

Argomenti

- **frameJSON : Collection**

è un oggetto JSON che contiene chiave e valore di inizializzazione dei campi dell'attributo info.

- **disableSaver : boolean**

se è uguale a false il contenuto di frame non viene salvato.

+ **initByDefault(disableSaver) : frame**

permette di inizializzare i campi dell'attributo info con i parametri di default.

Argomenti

- **disableSaver : boolean**

se è uguale a false il contenuto di frame non viene salvato.

+ **getJSON() : String**

restituisce la collection dell'attributo info.

+ **getType() : String**

restituisce la stringa frame per indicare che il tipo dell'oggetto è frame.

+ **getGOContent() : Collection**

restituisce il contenuto della collezione content che è un insieme di oggetti JSON che fanno parte del frame.

+ **selectGO(idGO) : GObject**

restituisce l'oggetto grafico con id = idGO contenuto all'interno del frame.

Argomenti

- **idGO : String**

valore dell'id dell'oggetto grafico da restituire.

+ deselectGO() : void

deseleziona l'oggetto grafico portando a null l'attributo selectedGo.

+ getSelectedGOType() : String

restituisce il tipo dell'oggetto grafico selezionato.

+ addGO(GO : Collection, type : String) : frame

aggiunge un oggetto di tipo type al frame e restituisce il riferimento del frame.

Argomenti

- **GO : Collection**

un oggetto JSON che serve per inizializzare l'oggetto da aggiungere al frame;

- **type : String**

contiene il tipo dell'oggetto da inserire nel frame.

+ removeSelectedGO(idGO : String) : frame

se l'oggetto con id = idGO è selezionato lo deselecta.

Argomenti

- **idGO : Collection**

rappresenta l'id dell'oggetto da selezionare.

+ getGOJSON(idGO : String) : Collection

restituisce l'oggetto JSON con id= idGO appartenente al frame.

Argomenti

- **idGO : String**

contiene l'id dell'oggetto che dev'essere restituito.

+ updateSelectedGO(update : Collection) : Collection

aggiorna i campi dell'oggetto selezionato.

Argomenti

- **update : Collection**

è un oggetto JSON che contiene chiave e valore dei campi da aggiornare dell'oggetto selezionato.

+ **resizeGO(resize : Collection, idGO : String) : void**

aggiorna i campi dell'oggetto con un determinato id, per permettere il ridimensionamento dell'oggetto.

Argomenti

- **resize : Collection**
è un oggetto JSON che contiene chiave e valore dei campi height, width, dataX e dataY per permettere di ridimensionare l'oggetto appartenente al frame;
- **idGO : String**
contiene l'id dell'oggetto da ridimensionare.

+ **dragGO(drag : Collection, idGO : String) : void**

aggiorna i campi dell'oggetto con un determinato id, per permettere lo spostamento dell'oggetto nel template grafico.

Argomenti

- **drag : Collection**
è un oggetto JSON che contiene chiave e valore dei campi dataX e dataY per permettere lo spostamento dell'oggetto appartenente al frame;
- **idGO : String**
contiene l'id dell'oggetto da spostare.

+ **updateGO(update : Collection, idGO : String) : void**

aggiorna i campi dell'oggetto con un determinato id appartenente al frame.

Argomenti

- **update : Collection**
è un oggetto JSON che contiene chiave e valore dei campi da aggiornare dell'oggetto con un determinato id;
- **idGO : String**
contiene l'id dell'oggetto da aggiornare.

+ **save() : void**

salva le operazioni pendenti nel database.

+ **getBackgroundColor() : String**

restituisce il colore in formato esadecimale dello sfondo del frame.

3.5.4 premi/client/editor/lib/GOContainer

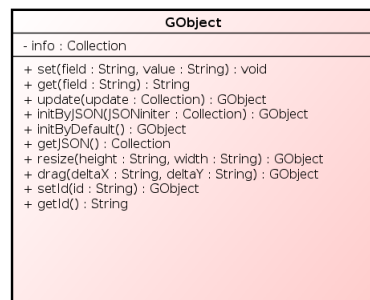


Figura 17: Diagramma della classe premi/client/editor/GOContainer

Descrizione

è una classe che rappresenta il contenitore degli oggetti che possono essere inseriti in un frame.

Classi ereditate

- GObject

Attributi

- info : Collection

l'attributo info è un oggetto JSON che estende l'attributo info ereditato da GObject. I campi aggiuntivi sono:

- *background*: è un oggetto JSON che contiene i seguenti campi:
 - *image*: definisce il percorso dell'immagine di background;
 - *size*: definisce la grandezza dell'immagine di background;
 - *color*: definisce il colore di background;
 - *repeat*: definisce il metodo di ripetizione dello sfondo di background;
 - *type*: definisce il nome del tipo dell'oggetto.
- *content*: è un oggetto JSON che contiene gli oggetti che fanno parte del frame;
- *idpres*: identifica l'id della presentazione a cui si riferisce.
- *owner*:

- **selectedGO : GObject**

contiene l'oggetto GObject contenuto nel frame corrente selezionato dall'utente.

Metodi

- **findNewId() : String**

trova un nuovo id valido per il frame corrente.

+ **set(field : String, value : String) : void**

permette di settare un campo dell'attributo info.

Argomenti

- **field : String**
field identifica il campo da settare di info;
- **value : String**
value rappresenta il valore del campo da settare su l'attributo info.

+ **get(field : String) : String**

restituisce il valore di un campo dell'attributo info.

Argomenti

- **field : String**
field identifica l'attributo info di cui si vuole venga restituito il valore.

+ **update(update : Collection) : frame**

permette di aggiornare i campi dell'attributo info.

Argomenti

- **update : Collection**
update è un oggetto JSON che contiene chiave e valore dei campi che devono essere aggiornati.

+ **initByJSON(frameJSON : Collection, disableSaver: boolean) : frame**

permette di inizializzare l'attributo info tramite un oggetto JSON.

Argomenti

- **frameJSON : Collection**
è un oggetto JSON che contiene chiave e valore di inizializzazione dei campi dell'attributo info.
- **disableSaver : boolean**
se è uguale a false il contenuto di frame non viene salvato.

+ **initByDefault(disableSaver) : frame**

permette di inizializzare i campi dell'attributo info con i parametri di default.

Argomenti

- **disableSaver : boolean**
se è uguale a false il contenuto di frame non viene salvato.

+ **getGOContent() : Collection**

restituisce il contenuto della collezione content che è un insieme di oggetti JSON che fanno parte del frame.

+ **selectGO(idGO) : GObject**

restituisce l'oggetto grafico con id = idGO contenuto all'interno del frame.

Argomenti

- **idGO : String**
valore dell'id dell'oggetto grafico da restituire.

+ **deselectGO() : void**

deseleziona l'oggetto grafico portando a null l'attributo selectedGo.

+ **getSelectedGOId() : String**

restituisce l'id dell'oggetto grafico selezionato.

+ **getSelectedGOType() : String**

restituisce il tipo dell'oggetto grafico selezionato.

+ **getSelectedGO() : String**

restituisce l'oggetto grafico selezionato.

+ setSelectedGO(concreteGObj : GObject) : frame

setta l'oggetto concreteGObj come oggetto selezionato.

Argomenti

- **concreteGObj : GObject**
oggetto GObject.

+ addGO(GO : Collection, type : String) : GOContainer

aggiunge un oggetto di tipo type al GOContainer e restituisce il riferimento del GOContainer.

Argomenti

- **GO : Collection**
un oggetto JSON che serve per inizializzare l'oggetto da aggiungere al GOContainer;
- **type : String**
contiene il tipo dell'oggetto da inserire nel GOContainer.

+ removeSelectedGO(idGO : String) : GOContainer

deseleziona l'oggetto con id = idGO se non è deselezionato.

Argomenti

- **idGO : Collection**
rappresenta l'id dell'oggetto da deselezionare.

+ getGOJSON(idGO : String) : Collection

restituisce l'oggetto JSON con id= idGO appartenente al frame.

Argomenti

- **idGO : String**
contiene l'id dell'oggetto che dev'essere restituito.

+ updateSelectedGO(update : Collection) : Collection

aggiorna i campi dell'oggetto selezionato.

Argomenti

- **update : Collection**

è un oggetto JSON che contiene chiave e valore dei campi da aggiornare dell'oggetto selezionato.

+ **resizeGO(resize : Collection, idGO : String) : void**

aggiorna i campi dell'oggetto con un determinato id, per permettere il ridimensionamento dell'oggetto.

Argomenti

- **resize : Collection**

è un oggetto JSON che contiene chiave e valore dei campi height, width, dataX e dataY per permettere di ridimensionare l'oggetto appartenente al frame;

- **idGO : String**

contiene l'id dell'oggetto da ridimensionare.

+ **dragGO(drag : Collection, idGO : String) : void**

aggiorna i campi dell'oggetto con un determinato id, per permettere lo spostamento dell'oggetto nel template grafico.

Argomenti

- **drag : Collection**

è un oggetto JSON che contiene chiave e valore dei campi dataX e dataY per permettere lo spostamento dell'oggetto appartenente al frame;

- **idGO : String**

contiene l'id dell'oggetto da spostare.

+ **updateGO(update : Collection, idGO : String) : void**

aggiorna i campi dell'oggetto con un determinato id appartenente al frame.

Argomenti

- **update : Collection**

è un oggetto JSON che contiene chiave e valore dei campi da aggiornare dell'oggetto con un determinato id;

- **idGO : String**

contiene l'id dell'oggetto da aggiornare.

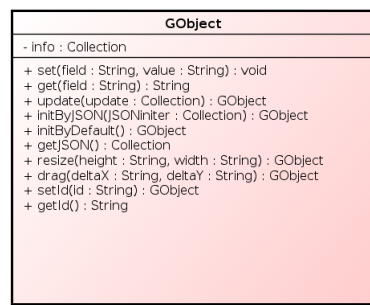


Figura 18: Diagramma della classe premi/client/editor/GObject

3.5.5 premi/client/editor/lib/Image

Descrizione

è una classe che rappresenta un oggetto immagine. Contiene i metodi per gestire un'immagine.

Classi ereditate

- GObject

Attributi

- info : Collection

l'attributo info è un oggetto JSON che estende l'attributo info ereditato da GObject. I campi aggiuntivi sono:

- *src*: definisce il percorso dell'immagine;
- *type*: identifica il tipo di oggetto.

- selectedGO : GObject

contiene l'oggetto GObject contenuto nel frame corrente selezionato dall'utente.

Metodi

+ set(field : String, value : String) : void

permette di settare un campo dell'attributo info.

Argomenti

- **field : String**
identifica il campo da settare di info;
- **value : String**
rappresenta il valore del campo da settare su l'attributo info.

+ **update(update : Collection) : image**

permette di aggiornare i campi dell'attributo info.

Argomenti

- **update : Collection**
update è un oggetto JSON che contiene chiave e valore dei campi che devono essere aggiornati.

+ **initByJSON(imageJSON : Collection) : GObject**

permette di inizializzare l'attributo info tramite un oggetto JSON.

Argomenti

- **imageJSON : Collection**
è un oggetto JSON che contiene chiave e valore di inizializzazione dei campi dell'attributo info.

+ **initByDefault() : GObject**

permette di inizializzare i campi dell'attributo info con i parametri di default.

+ **setImage(src : String) : GObject**

setta l'url(percorso) dell'immagine.

Argomenti

- **src : String**
identifica l'url(percorso) dell'immagine.

+ **getImage() : String**

restituisce l'url(percorso) dell'immagine.

+ **getType() : String**

restituisce la stringa image per identificare che il tipo dell'oggetto è image.

3.5.6 premi/client/editor/lib/Infographic

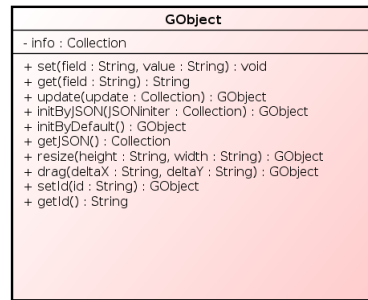


Figura 19: Diagramma della classe premi/client/editor/GObject

Descrizione

è una classe che rappresenta un oggetto infografica. Un infografica contiene i frame e gli oggetti che si vogliono visualizzare nella presentazione.

Classi ereditate

- GOContainer

Attributi

- info : Collection

l'attributo info è un oggetto JSON che estende l'attributo info ereditato da GObject. I campi aggiuntivi sono:

- *framesId*: array che contiene gli id dei frame appartenenti all'infografica;
- *type*: identifica il tipo di oggetto ovvero infografica.

- frames : Collection

contiene degli oggetti JSON che rappresentano i frame appartenenti all'infografica.

- saver : Saver

oggetto Saver che si occupa delle modifiche e dei salvataggi su database.

Metodi

- **removeFrame(idFrame : String) : void**

rimuove un frame dall'infografica

Argomenti

- **idFrame : String**

Identifica l'id del frame da rimuovere dall'infografica.

+ **update(update : Collection) : image**

permette di aggiornare i campi dell'attributo info.

Argomenti

- **update : Collection**

update è un oggetto JSON che contiene chiave e valore dei campi che devono essere aggiornati.

+ **initByJSON(imageJSON : Collection) : GObject**

permette di inizializzare l'attributo info tramite un oggetto JSON.

Argomenti

- **imageJSON : Collection**

è un oggetto JSON che contiene chiave e valore di inizializzazione dei campi dell'attributo info.

+ **initByDefault() : GObject**

permette di inizializzare i campi dell'attributo info con i parametri di default.

+ **setImage(src : String) : GObject**

setta l'url(percorso) dell'immagine.

Argomenti

- **src : String**

identifica l'url(percorso) dell'immagine.

+ getImage() : String

restituisce l'url(percorso) dell'immagine.

+ getType() : String

restituisce la stringa image per identificare che il tipo dell'oggetto è image.

+ initByJSON(infJSON : Collection,framesId : String[],frames : Collection) : infographic

permette di inizializzare l'oggetto infografica

Argomenti

- **infJSON : Collection**
è un oggetto JSON che contiene chiave e valore di inizializzazione dei campi dell'attributo info;
- **framesId : String[]**
array di identificativi che rappresenta gli id dei frame da aggiungere all'infografica;
- **frames : Collection**
contiene oggetti JSON che identificano i frame appartenenti all'infografica.

+ initByDefault() : infographic

permette di inizializzare i campi dell'attributo info con i parametri di default e di lasciare vuoti l'attributo framesId e frames.

+ getType() : String

restituisce la stringa infographic perchè il tipo di oggetto è un infografica.

+ getFramesContent() : Collection

restituisce gli oggetti JSON che rappresentano i frame appartenenti all'infografica.

+ getFramesId() : String

restituisce un array in cui ciascun elemento identifica un frame appartenente all'infografica.

+ getSelectedGO() : String

restituisce l'oggetto selezionato nell'infografica.

+ setSelectedGO(concreteGObj : Collection) : infographic

imposta su selezionato un oggetto dell'infografica.

Argomenti

- **concreteGObj : Collection**
rappresenta l'oggetto JSON da selezionare nell'infografica.

+ selectedGO(idGO : String) : infographic

seleziona un oggetto dell'infografica.

Argomenti

- **idGO : String**
identifica l'id dell'oggetto da selezionare.

+ addGO(GO : Collection, type : String) : infographic

aggiunge un oggetto di tipo type all'infografica e restituisce il riferimento dell'infografica.

Argomenti

- **GO : Collection**
un oggetto JSON che serve per inizializzare l'oggetto da aggiungere all'infografica;
- **type : String**
contiene il tipo dell'oggetto da inserire nell'infografica.

+ removeSelectedGO(idGO : String) : infographic

se l'oggetto con id = idGO è selezionato lo deselecta.

Argomenti

- **idGO : Collection**
rappresenta l'id dell'oggetto da selezionare.

+ getGOJSON(idGO : String) : Collection

restituisce l'oggetto JSON con id= idGO appartenente all'infografica.

Argomenti

- **idGO : String**
contiene l'id dell'oggetto che dev'essere restituito.

+ **updateSelectedGO(update : Collection) : infographic**

aggiorna i campi dell'oggetto selezionato.

Argomenti

- **update : Collection**
è un oggetto JSON che contiene chiave e valore dei campi da aggiornare dell'oggetto selezionato.

+ **updateGO(update : Collection, idGO : String) : infographic**

aggiorna i campi di un oggetto dell'infografica.

Argomenti

- **update : Collection**
è un oggetto JSON che contiene chiave e valore dei campi da modificare dell'oggetto da aggiornare;
- **idGO : String**
identifica l'id dell'oggetto da aggiornare.

+ **resizeGO(resize : Collection, idGO : String) : void**

aggiorna i campi dell'oggetto con un determinato id, per permettere il ridimensionamento dell'oggetto.

Argomenti

- **resize : Collection**
è un oggetto JSON che contiene chiave e valore dei campi height, width, dataX e dataY per permettere di ridimensionare l'oggetto appartenente al frame;
- **idGO : String**
contiene l'id dell'oggetto da ridimensionare.

+ **dragGO(drag : Collection, idGO : String) : void**

aggiorna i campi dell'oggetto con un determinato id, per permettere lo spostamento dell'oggetto nel template grafico.

Argomenti

- **drag : Collection**

è un oggetto JSON che contiene chiave e valore dei campi dataX e dataY per permettere lo spostamento dell'oggetto appartenente al frame;

- **idGO : String**

contiene l'id dell'oggetto da spostare.

+ **hasFrame(idGO : String) : Collection**

restituisce le proprietà in JSON di un oggetto dell'infografica.

Argomenti

- **idGO : String**

rappresenta l'id dell'oggetto dell'infografica che si vuole restituire.

+ **save() : void**

salva le operazioni pendenti nel database.

3.5.7 premi/client/editor/lib/interactInit

Descrizione

classe che contiene i metodi per la gestione del ridimensionamento e dello spostamento degli oggetti.

Attributi

- **observer : Observer**

oggetto Observer che viene utilizzato per osservare un oggetto grafico;

- **restrictArea : String**

identifica l'area in cui un oggetto può essere spostato.

Metodi

- **initializeGO(idwrapperGO : String) : void**

inizializza un oggetto grafico impostando l'area di restrizione e impostando l'observer sull'oggetto. Si appoggia alla libreria interact e sulla classe observer.

Argomenti

- **idwrapperGO : String**
Identifica l'id dell'oggetto da inizializzare.

+ **setObserver(conObserver : Observer) : interactInit**

imposta un oggetto Observer sull'oggetto da controllare. restituisce un riferimento dell'oggetto interactInit.

Argomenti

- **conObserver : Observer**
Identifica l'observer da impostare sull'oggetto.

+ **setRestrictArea(area : String) : interactInit**

imposta l'area di restrizione entro cui un oggetto può essere spostato. Restituisce un riferimento dell'oggetto interactInit.

Argomenti

- **area : String**
Identifica l'area di restrizione.

+ **initializeText(idwrappergo : String) : interactInit**

inizializza un determinato oggetto di tipo text. Restituisce un riferimento dell'oggetto interactInit.

Argomenti

- **idwrappergo : String**
Identifica l'id dell'oggetto text da inizializzare.

+ **initializeShape(idwrappergo : String) : interactInit**

inizializza un determinato oggetto di tipo shape impostando il comportamento per lo spostamento e ridimensionamento dell'oggetto. Restituisce un riferimento dell'oggetto interactInit.

Argomenti

- **idwrappergo : String**
Identifica l'id dell'oggetto shape da inizializzare.

+ initializeFrame(idwrappergo : String) : interactInit

inizializza un determinato oggetto di tipo frame. Restituisce un riferimento dell'oggetto interactInit.

Argomenti

- **idwrappergo : String**
Identifica l'id dell'oggetto frame da inizializzare.

+ initializeImage(idwrappergo : String) : interactInit

inizializza un determinato oggetto di tipo image impostando i comportamenti per lo spostamento e il ridimensionamento dell'oggetto. Restituisce un riferimento dell'oggetto interactInit.

Argomenti

- **idwrappergo : String**
Identifica l'id dell'oggetto image da inizializzare.

+ initializeInf(id : String) : interactInit

inizializza un determinato oggetto di tipo infografica. Restituisce un riferimento dell'oggetto interactInit.

Argomenti

- **idwrappergo : String**
Identifica l'id dell'oggetto infografica da inizializzare.

+ unSet(idwrappergo : String) : interactInit

disabilita interactjs per un determinato oggetto togliendo la possibilità di ridimensionamento e spostamento dell'oggetto.

Argomenti

- **idwrappergo : String**
Identifica l'id dell'oggetto su cui disabilitare interactjs.

3.5.8 premi/client/client/lib/Observer

Descrizione

è la classe che si occupa di osservare degli oggetti grafici impostando e inviando dei segnali.

Attributi

- slots : Collection

array di oggetti JSON in cui la chiave rappresenta un segnale e il valore l'azione da intraprendere.

Metodi

+ on(signal : String, func : String)

assegna la funzione func al segnale signal.

Argomenti

- **signal : String**
identifica il nome del segnale;
- **func : String**
identifica la funzione da eseguire al verificarsi del segnale;

+ emit(signal : String, param1 : String, param2 : String, param3 : String, param4 : String)

assegna i parametri ad un determinato segnale.

Argomenti

- **signal : String**
identifica il nome del segnale;
- **param1,param2,param3,param4 : String**
identificano i parametri da attribuire al segnale signal;

3.5.9 premi/client/editor/lib/saver

Descrizione

rappresenta un oggetto che permette ad un contenitore di oggetti di interfacciarsi con il database ed effettuare le modifiche.

Attributi

- **containerId : String**

identifica l'id dell'oggetto da interfacciare con il database;

- **containerType : String**

identifica il tipo dell'oggetto da interfacciare con il database;

- **updates : Collection**

oggetto JSON che contiene i seguenti campi:

- *image:*

Metodi

+ void operation0()

Descrizione del metodo

Argomenti

- **nomeArgomento : tipoArgomento**

Descrizione argomento

Note

- Deve essere esplicitamente marcato come costante (?)
- Deve possedere qualche caratteristica
- Metodo ridefinito

3.5.10 premi/client/editor/lib/Shape

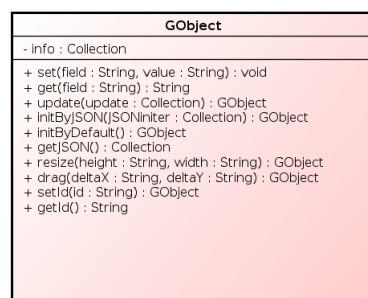


Figura 20: Diagramma della classe premi/client/editor/GObject

Descrizione

è una classe che rappresenta un oggetto shape. Contiene i metodi per gestire uno shape.

Classi ereditate

- GObject

Attributi

- info : Collection

l'attributo info è un oggetto JSON che estende l'attributo info ereditato da GObject. I campi aggiuntivi sono:

- *src*: definisce il percorso dello shape da visualizzare. Ogni shape è rappresentato da un'immagine svg;
- *content*: definisce il contenuto dello shape;
- *fill*: definisce ;
- *stroke*: definisce ;
- *strokeWidth*: definisce ;
- *type*: identifica il tipo di oggetto.

Metodi

+ **get(field : String) : String**

restituisce una proprietà dell'oggetto shape.

Argomenti

- **field : String**
identifica la proprietà da restituire;

+ **set(field : String, value : String) : shape**

permette di settare un campo dell'attributo info.

Argomenti

- **field : String**
identifica il campo da settare di info;
- **value : String**
rappresenta il valore del campo da settare su l'attributo info.

+ **update(update : Collection) : shape**

permette di aggiornare i campi dell'attributo info.

Argomenti

- **update : Collection**
update è un oggetto JSON che contiene chiave e valore dei campi che devono essere aggiornati.

+ **initByJSON(shapeJSON : Collection) : shape**

permette di inizializzare l'attributo info tramite un oggetto JSON.

Argomenti

- **shapeJSON : Collection**
è un oggetto JSON che contiene chiave e valore di inizializzazione dei campi dell'attributo info.

+ **initByDefault() : shape**

permette di inizializzare i campi dell'attributo info con i parametri di default.

+ **setShape(src : String) : GObject**

setta l'url(percorso) dello shape.

Argomenti

- **src : String**
identifica l'url(percorso) dello shape.

+ **getShape() : String**

restituisce l'url(percorso) dello shape.

+ **getType() : String**

restituisce la stringa shape per identificare che il tipo dell'oggetto è image.

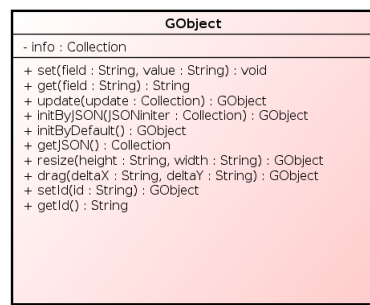


Figura 21: Diagramma della classe premi/client/editor/GObject

3.5.11 premi/client/editor/lib/Text

Descrizione

è una classe che rappresenta un oggetto text. Contiene i metodi per gestire uno text.

Classi ereditate

- GObject

Attributi

- info : Collection

l'attributo info è un oggetto JSON che estende l'attributo info ereditato da GObject. I campi aggiuntivi sono:

- *text*: definisce il testo da visualizzare sull'oggetto;
- *colorText*: definisce il colore del testo;
- *weight*: definisce come visualizzare il testo (es Bold, normal);
- *fontFamily*: definisce il font del testo;
- *textDecoration*: definisce particolari decorazioni da attribuire al testo;
- *sizeFontText*: definisce la grandezza del testo;
- *type*: identifica il tipo di oggetto.

Metodi

+ get(field : String) : String

restituisce una proprietà dell'oggetto text.

Argomenti

- **field : String**
identifica la proprietà da restituire;

+ **set(field : String, value : String) : shape**

permette di settare un campo dell'attributo info.

Argomenti

- **field : String**
identifica il campo da settare di info;
- **value : String**
rappresenta il valore del campo da settare su l'attributo info.

+ **update(update : Collection) : shape**

permette di aggiornare i campi dell'attributo info.

Argomenti

- **update : Collection**
update è un oggetto JSON che contiene chiave e valore dei campi che devono essere aggiornati.

+ **initByJSON(shapeJSON : Collection) : shape**

permette di inizializzare l'attributo info tramite un oggetto JSON.

Argomenti

- **shapeJSON : Collection**
è un oggetto JSON che contiene chiave e valore di inizializzazione dei campi dell'attributo info.

+ **initByDefault() : shape**

permette di inizializzare i campi dell'attributo info con i parametri di default.

+ **setText(text : String) : GObject**

setta il testo del text.

Argomenti

- **text : String**
identifica il testo da inserire.

+ **getText() : String**

restituisce il testo dell'oggetto.

+ **getType() : String**

restituisce la stringa shape per identificare che il tipo dell'oggetto è image.

3.6 premi/client/frameEditor

3.7 premi/client/infographicEditor

3.8 premi/client/trailsEditor

3.9 premi/client/userManager

3.10 premi/client/viewer

4 Tracciamento

5 Diagrammi di Sequenza