09-06-2015

Definizione di Prodotto

Informazioni sul documento

Nome Documento	Definizione di Prodotto
Versione	1.0.0
Stato	*Formale*
Uso	*Esterno*
Data Creazione	09-06-2015
Data Ultima Modifica	09-06-2015
Redazione	Fossa Manuel, Venturelli Giovanni, Tollot Pietro, Busetto Matteo
Approvazione	Busetto Matteo
Verifica	Petrucci Mauro
Lista distribuzione	*LateButSafe* Prof. Tullio Vardanega Prof. Riccardo Cardin Proponente Zucchetti S.p.a.

Sommario

Il presente documento riporta la Definizione di Prodotto effettuata per il capitolato Premi.

Registro delle modifiche

Tab 1: Versionamento del documento

Versione	Autore	Data	Descrizione
0.7.0	Busetto Matteo	24-06-2015	Aggiunta di contenuti. Inserimento del capitolo Package::Premi::Controller
0.5.0	Venturelli Giovanni	20-06-2015	Aggiunta di contenuti. Inserimento del capitolo Package::Premi::Model
0.4.0	Fossa Manuel	15-06-2015	Aggiunta di contenuti. Inserimento del capitolo Package::Premi::View
0.3.0	Tollot Pietro	12-06-2015	Aggiunta di contenuti. Inserimento del capitolo Standard di Progetto
0.2.5	Gabelli Pietro	09-06-2015	Aggiunta di contenuti. Inserimento del capitolo Introduzione e Descrizione generale
0.1.0	Gabelli Pietro	08-06-2015	Stesura dello scheletro del documento

Storico

RP -> RQ

Versione 1..0.0	**Nominativo**
Redazione	Fossa Manuel, Venturelli Giovanni, Tollot Pietro, Busetto Matteo
Verifica	Petrucci Mauro
Approvazione	Busetto Matteo

Tab 2: Storico ruoli RP -> RQ

Indice

Elenco delle figure

Elenco delle tabelle

1 Introduzione

1.1 Scopo del documento

Il presente documento descrive la progettazione di dettaglio definita per il progetto Premi. Il documento si basa sulla SpecificaTecnica_v.1.0.0.pdf. I programmatori si serviranno di tale documento per procedere con le attività di codifica.

1.2 Scopo del Prodotto

Lo scopo del Progetto$_g$ è la realizzazione un Software$_g$ per la creazione ed esecuzione di presentazioni multimediali favorendo l'uso di tecniche di storytelling e visualizzazione non lineare dei contenuti.

1.3 Glossario

Al fine di evitare ogni ambiguità di linguaggio e massimizzare la comprensione dei documenti, i termini tecnici, di dominio, gli acronimi e le parole che necessitano di essere chiarite sono riportate nel documento Glossario_v.2.0.0.pdf. Ogni occorrenza di vocaboli presenti nel Glossario è marcata da una "g" minuscola in pedice.

1.4 Riferimenti

1.4.1 Normativi

- Regole del Progetto$_g$ didattico, reperibili all'Indirizzo$_g$:
  http://www.math.unipd.it/~tullio/IS-1/2014/Progetto/PD01.pdf
- Vincoli di organigramma, consultabili all'Indirizzo$_g$:
  http://www.math.unipd.it/~tullio/IS-1/2014/Progetto/PD01b.html
- Norme di Progetto$_g$: NormeDiProgetto_v.2.0.0.pdf;
- Specifica Tecnica$_g$: SpecificaTecnica_v.1.0.0.pdf;
- Capitolato d'appalto C4: Premi: Software$_g$ 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$_g$ modulo A:
  - Il ciclo di vita$_g$ del Software$_g$;
  - Gestione di Progetto$_g$.

  http://www.math.unipd.it/~tullio/IS-1/2014/ ;
- Ingegneria del Software$_g$ - Ian Sommerville - 9a Edizione (2010).

2 Descrizione generale

Il sistema si pone come obbiettivo quello di permettere la creazione di presentazioni efficaci dal punto di vista dello storytelling anche ad utenti non esperti.
L'applicazione Premi permette all'utente di creare ed eseguire presentazioni personalizzate. Attraverso la creazione di Frame$_g$ e la loro modellazione, l'utente potrà definire un Percorso$_g$ di presentazione lineare oppure più percorsi$_g$ che prevedono la possibilità di scegliere con quale continuare il flusso di esecuzione. Questo significa che il Percorso$_g$ di Frame$_g$ che sarà visualizzato sarà scelto dal presentatore in fase di visualizzazione.
L'applicazione è strutturata in modo gerarchico, ossia ogni Frame$_g$ ha almeno un padre (tranne la radice che può avere solamente figli). Questo permette di creare presentazioni strutturate a livelli, rendendo molto semplice la possibilità, durante l'esecuzione, di saltare determinati rami della presentazione. Inoltre, l'utente potrà inserire Bookmark$_g$ i quali permettono di saltare ad un Frame$_g$ padre in modo semplice e veloce.
Il sistema permetterà di creare e modificare presentazioni se connessi alla rete mentre l'utente potrà eseguire le proprie presentazione anche offline a patto di averle precedentemente scaricate dal Server$_g$. Il sistema sarà implementato utilizzando tecnologie WEB$_g$ che lo renderanno altamente portabile.

2.1 funzioni$_g$ del prodotto

Il prodotto offre un'interfaccia WEB$_g$ che permetterà di:

- Registrarsi, accedere al proprio Account$_g$ ed effettuare il Logout$_g$;
- Gestire il proprio Account$_g$;
- Creare una nuova presentazione da dispositivo Desktop$_g$;
- Modificare una presentazione da dispositivo Desktop$_g$;
- Modificare parzialmente la presentazione da dispositivo mobile$_g$;
- Eseguire una presentazione salvata sul proprio Account$_g$;
- Eseguire una presentazione locale;
- Creare Infografiche$_g$ a partire da una presentazione;
- Modificare Infografiche$_g$ create;
- Gestire il proprio archivio di File$_g$ media;
- Scaricare una presentazione in locale.

2.2 Caratteristiche degli utenti

Il prodotto si rivolge a qualsiasi tipo di utente interessato ad una facile creazione e modellazione di presentazioni ed Infografiche$_g$. Non emergono quindi restrizioni particolari riguardo le caratteristiche dell'utenza.

2.3 Vincoli generali

Per poter utilizzare il Software$_g$ Premi è necessario disporre di un computer o di un dispositivo mobile$_g$ con installato almeno uno dei principali Browser$_g$ quali Google Chrome, Mozilla Firefox e Safari scaricabili dai rispettivi siti ufficiali. Inoltre, il dispositivo deve supportare le seguenti tecnologie:

- HTML5;
- CSS3;
- Javascript.

Il prodotto non richiede particolari Requisiti$_g$ hardware anche se gli stessi possono influenzarne la velocità di esecuzione.

3 Standard di progetto

3.1 Standard di progettazione architetturale

Gli standard di progettazione architetturale sono definiti nel documento SpecificaTecnica_v.1.0.0.pdf.

3.2 Standard di documentazione del codice

Gli standard per la scrittura di documentazione del codice sono definiti nelle NormeDiProgetto_v.2.0.0.pdf

3.3 Standard di denominazione di entità e relazioni

Tutti gli elementi (package, classi, metodi o attributi) definiti, devono avere denominazioni chiare ed autoesplicative. Nel caso il nome risulti lungo, è preferibile preferire la chiarezza alla lunghezza.
Sono ammesse abbreviazioni se:

- immediatamente comprensibili;
- non ambigue;
- sufficientemente contestualizzate.

Le regole tipografiche relative ai nomi delle entità sono definite nelle NormeDiProgetto_v.2.0.0.pdf.

3.4 Standard di programmazione

Gli standard di programmazione sono definiti e descritti nelle NormeDiProgetto_v.2.0.0.pdf.

3.5 Strumenti di lavoro

Gli strumenti da adottare e le procedure per utilizzarli correttamente durante la realizzazione del prodotto software sono definiti nelle NormeDiProgetto_v.2.0.0.pdf.

4 NodeServer

Il seguente diagramma delle classi è stato esteso con le primitive:

- **«Resource»** : rappresenta una risorsa associata ad un certo url a cui sono disponibili dei servizi
- **«Node»** : rappresenta una parte di url a cui non sono disponibili servizi ma è utile per suddividere quest'ultimi
- **«Server»** : rappresenta la radice dei servizi offerti dal server
- **«Path»** : indica una aggiunta in coda all' url attuale per raggiungere una nuova risorsa o nodo
- **«Middleware»** : indica un middleware, un insieme di funzionalità chiamate ogni qualvolta si accede a risorse attraversando questo elemento

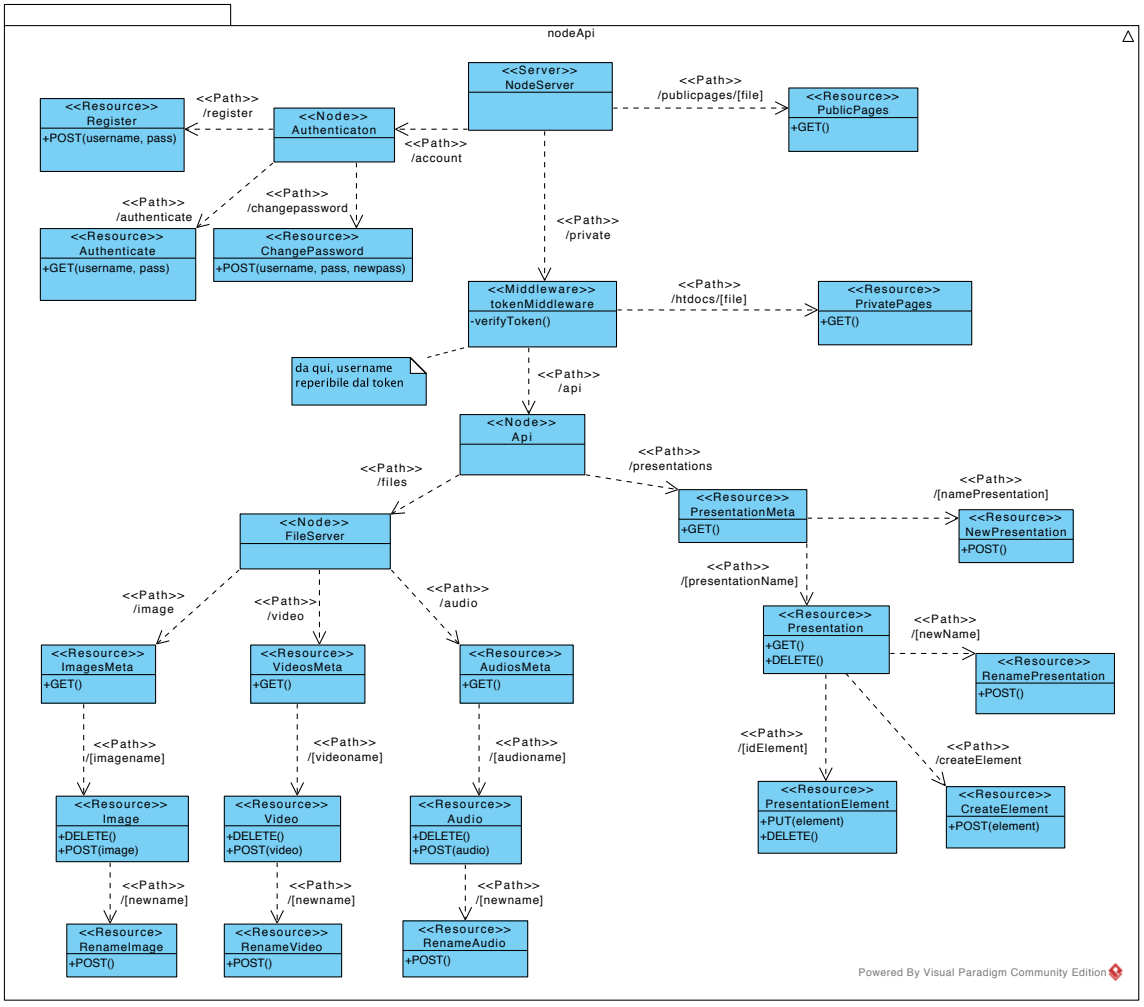

Fig 1: Servizi RESTfull offerti dal server nodeJs

4.1 Risorse e Servizi

- **NodeServer:** radice dei servizi offerti dal server node:
  1. server per pagine html, css e javascript associati
  2. servizi di autenticazione stateless
  3. file server per salvare sul server file statici multimediali(immagini, audio, video), richiedono autenticazione
  4. servizi di interazione con un database MongoDB dove sono persistentemente salvate le presentaizoni, richiedono autenticazione
- **Register:**
  - /account/register POST
    - **descrizione:** verifica la presenza dello username nella collezione in mongoDB relativa agli account, se non è già presente inserisce username e password ricevuti altrimenti annulla l'operazione di inserimento e ritorna un messaggio di errore
    - **parametri input body(application/json):** username : string, password : string
    - **parametri output body(application/json):** success : boolean, message : string
- **Authenticate:**
  - /account/authenticate GET
    - **descrizione:** verifica username e password ricevuti e se presenti nella opportuna collezione in mongoDB ritorna un token per l'accesso ai servizi protetti, altrimenti ritorna solo un messaggio d'errore
    - **parametri input header:** username : string, password : string
    - **parametri output body(application/json):** success : boolean, message : string, token : string
- **ChangePassword:**
  - /account/changepassword POST
    - **descrizione:** verifica username e password ricevuti e se presenti nella opportuna collezione in mongoDB e modifica la password con newpassword, altrimenti ritorna solo un messaggio d'errore
    - **parametri input body(application/json):** username : string, password : string, new password : string
    - **parametri output body(application/json):** success : boolean, message : string
- **PublicPages:**
  - /publicpages/[file] GET
    - **parametri input body:** /

    - **output body:** fileStatico

- **tokenMiddleware:** verifica che il token passato nel campo Authorization dell' Header sia valido, ne estrae lo username dell'utente e permette l'accesso ai servizi richiesti
- **PrivatePages:**
  - /private/htdocs/[file] GET
    - **descrizione:** se presente il file [file] nella cartella /private/htdocs del server ritorna il file stesso
    - **parametri input header:** token : string
    - **output body:** fileStatico
- **PresentationMeta:**
  - /private/api/presentations GET
    - **descrizione:** cerca in mongoDB nella collezione associata alle presentazioni dell'utente, ritorna un array i cui elementi sono array associativi con le meta-informazioni riguardanti le presentazioni
    - **parametri input header:** token : string
    - **output body(application/json):** success : boolean, message : string, presentationMetas : array
- **NewPresentation:**
  - /private/api/presentations/[presentationName] POST
    - **descrizione:** crea una nuova presentazione con il nome [presentatioNname] se il nome non è già stato usata per un'altra presentazione dello stesso utente
    - **parametri input header:** token : string
    - **output body(application/json):** success : boolean, message : string
- **Presentation:**
  - /private/api/presentations/[presentationName] GET
    - **descrizione:** recupera la presentazione se esistente associata al nome passato come ultima parte dell'url
    - **parametri input header:** token : string
    - **output body(application/json):** success : boolean, message : string, presentation : object
  - /private/api/presentations/[presentationName] DELETE
    - **descrizione:** elimina la associazione dell'utente con nome il valore di [presentationName]
    - **parametri input header:** token : string
    - **output body(application/json):** success : boolean, message : string
- **RenamePresentation:**

  - /private/api/presentations/[presentationName]/[newname] POST
    - **descrizione:** rinomina la presentazione con il nome [presentatioNname] con il nome [newname]
    - **parametri input header:** token : string
    - **output body(application/json):** success : boolean, message : string

- **CreateElement:**
  - /private/api/presentations/[presentationName]/[createElementPOST
    - **descrizione:** crea nella presentazione [presentationName] dell'utente un nuovo elemento, ovvero l'oggetto element passato in input nel corpo della chiamata
    - **parametri input header:** token : string
    - **input body(application/json):** element : object
    - **output body(application/json):** success : boolean, message : string

- **PresentationElement:**
  - /private/api/presentations/[presentationName]/[idElement] PUT
    - **descrizione:** sostituisce nella presentazione dell'utente l'elemento con identificativo [idElement] con l'oggetto passato nel corpo in formato son
    - **parametri input header:** token : string
    - **input body(application/json):** element : object
    - **output body(application/json):** success : boolean, message : string
  - /private/api/presentations/[presentationName]/[idElement] DELETE
    - **descrizione:** elimina dalla presentazione con nome [presentationName] l'elemento con identificativo [idElement]
    - **parametri input header:** token : string
    - **output body(application/json):** success : boolean, message : string

- **ImagesMeta:**
  - /private/api/files/image GET
    - **descrizione:** ritorna un array con oggetti rappresentanti informazioni sui file immagine dell'utente sul server
    - **parametri input header:** token : string
    - **output body(application/json):** success : boolean, message : string, imageMetas : array

- **ImagesMeta:**
  - /private/api/files/image GET
    - **descrizione:** ritorna un array con oggetti rappresentanti informazioni sui file immagine dell'utente sul server
    - **parametri input header:** token : string

    * **output body(application/json):** success : boolean, message : string, imageMetas : array

- **Image:**
  - /private/api/files/image/[imagename] POST
    * **descrizione:** caricare da locale un nuovo file immagine nella cartella /users/[username]/images
    * **parametri input header:** token : string
    * **input body(multipart/form-data):** file
    * **output body(application/json):** success : boolean, message : string
  - /private/api/files/image/[imagename] DELETE
    * **descrizione:** elimina il file immagine [imagename] dalla cartella /users/[username]/images nel server
    * **parametri input header:** token : string
    * **output body(application/json):** success : boolean, message : string
- **RenameImage:**
  - /private/api/files/image/[imagename]/[newname] POST
    * **descrizione:** rinomina il file immagine [imagename] con il nuovo nome passato come valore di [newname] nella cartella /users/[username]/images
    * **parametri input header:** token : string
    * **output body(application/json):** success : boolean, message : string
- **VideosMeta:**
  - /private/api/files/video GET
    * **descrizione:** ritorna un array con oggetti rappresentanti informazioni sui file video dell'utente sul server
    * **parametri input header:** token : string
    * **output body(application/json):** success : boolean, message : string, videoMetas : array
- **Video:**
  - /private/api/files/video/[videoname] POST
    * **descrizione:** caricare da locale un nuovo file video nella cartella /users/[username]/videos
    * **parametri input header:** token : string
    * **input body(multipart/form-data):** file
    * **output body(application/json):** success : boolean, message : string
  - /private/api/files/video/[videoname] DELETE
    * **descrizione:** elimina il file video [videoname] dalla cartella /users/[username]/videos nel server

    - **parametri input header:** token : string
    - **output body(application/json):** success : boolean, message : string

- **RenameImage:**
  - /private/api/files/video/[videoname]/[newname] POST
    - **descrizione:** rinomina il file video [videoname] con il nuovo nome passato come valore di [newname] nella cartella /users/[username]/videos
    - **parametri input header:** token : string
    - **output body(application/json):** success : boolean, message : string

- **AudiosMeta:**
  - /private/api/files/audio GET
    - **descrizione:** ritorna un array con oggetti rappresentanti informazioni sui file audio dell'utente sul server
    - **parametri input header:** token : string
    - **output body(application/json):** success : boolean, message : string, audioMetas : array

- **Audio:**
  - /private/api/files/audio/[audioname] POST
    - **descrizione:** caricare da locale un nuovo file video nella cartella /users/[username]/videos
    - **parametri input header:** token : string
    - **input body(multipart/form-data):** file
    - **output body(application/json):** success : boolean, message : string
  - /private/api/files/audio/[audioname] DELETE
    - **descrizione:** elimina il file audio [audioname] dalla cartella /users/[username]/audios nel server
    - **parametri input header:** token : string
    - **output body(application/json):** success : boolean, message : string

- **RenameAudio:**
  - /private/api/files/audio/[audioname]/[newname] POST
    - **descrizione:** rinomina il file video [videoname] con il nuovo nome passato come valore di [newname] nella cartella /users/[username]/audios
    - **parametri input header:** token : string
    - **output body(application/json):** success : boolean, message : string

5 Package Premi::Model

Tutti i package seguenti appartengono al package Premi, quindi per ognuno di essi lo scope sarà: Premi::[nome package].

Tipo, obiettivo e funzione del componente: classe astratta, base delle classi usate per rappresentare gli elementi della presentazione.
Relazioni d'uso di altre componenti: è in relazione con il package Controller e con NodeAPI.

5.1 Classe SlideShowElements

Funzione
Classe astratta, base delle classi usate per rappresentare gli elementi della presentazione.
Scope
Model::SlideShow::SlideShowElements.
Utilizzo
Contiene gli attributi e i metodi comuni degli oggetti che rappresentano gli elementi della presentazione.
Attributi

- **id**
  - **Accesso**: Private;
  - **Tipo**: Integer;
  - **Descrizione**: indica l'identificativo univoco dell'elemento.
- **yIndex**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle y dell'elemento rispetto alla presentazione.
- **xIndex**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle x dell'elemento rispetto alla presentazione.
- **rotation**
  - **Accesso**: Private;
  - **Tipo**: Double;

  - **Descrizione**: rappresenta il grado di rotazione dell'oggetto.

- **height**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta l'altezza dell'oggetto.

- **width**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la larghezza dell'oggetto.

Metodi

- **setSize**(newHeight:double, newWidth:double)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta i campi height e width dell'oggetto.

- **setPosition**(posX:double, posY:double)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta i campi xIndex e yIndex dell'oggetto.

- **getSize**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Array;
  - **Descrizione**: restituisce un array contenente i valori di height e width.

- **getWidth**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Array;
  - **Descrizione**: restituisce un array contenente i valori di xIndex e yIndex.

- **getRotation**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Double;
  - **Descrizione**: restituisce il valore di rotation.

Ereditata da:

- Text (§5.1.1);
- Image (§5.1.2);
- Frame (§5.1.3);
- SVG (§5.1.4);
- Background (§5.1.7);
- Audio (§5.1.5);
- Video (§5.1.6).

5.1.1 Classe Text

Funzione
Classe concreta, i suoi elementi rappresentano un oggetto di tipo testo.
Scope
Model::SlideShow::SlideShowElements::Text.
Utilizzo
Il costruttore viene invocato da Inserter::insertText().
Attributi

- **font**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta il font dell'oggetto.
- **content**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta il contenuto del testo.
- **color**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta il colore dell'oggetto.

Metodi

- **Text**(id:integer, posX:double, posY:double, degrees:double)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto, imposta i campi id, xIndex, yIndex, rotation.

- **setFont**(newFont:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta il campo content dell'oggetto.
- **setColor**(newColor:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta il campo color dell'oggetto.
- **getFont**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: String;
  - **Descrizione**: restituisce il valore di font.
- **getColor**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: String;
  - **Descrizione**: restituisce il valore di color.
- **getContent**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: String;
  - **Descrizione**: restituisce il valore di content.

5.1.2 Classe Image

Funzione
Classe concreta, i suoi elementi rappresentano un oggetto di tipo immagine.
Scope
Model::SlideShow::SlideShowElements::Image.
Utilizzo
Il costruttore viene invocato da Inserter::insertImage().
Attributi

- **url**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta il percorso dell'oggetto.

Metodi

- **Image**(id:integer, posX:double, posY:double, degrees:double, ref:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto, imposta i campi id, xIndex, yIndex, rotation e url.
- **getUrl**()
  - **Accesso**: Public;
  - **Tipo**: String;
  - **Descrizione**: restituisce il valore di url.

5.1.3 Classe Frame

Funzione
Classe concreta, i suoi elementi rappresentano un oggetto di tipo frame.
Scope
Model::SlideShow::SlideShowElements::Frame.
Utilizzo
Il costruttore viene invocato da Inserter::insertFrame().
Attributi

- **prev**
  - **Accesso**: Private;
  - **Tipo**: Integer;
  - **Descrizione**: rappresenta l'id del frame precedente.
- **next**
  - **Accesso**: Private;
  - **Tipo**: Integer;
  - **Descrizione**: rappresenta l'id del frame successivo.
- **bookmark**
  - **Accesso**: Private;
  - **Tipo**: Bool;
  - **Descrizione**: è a 1 se il frame è un bookmark, 0 altrimenti.
- **choices**
  - **Accesso**: Private;
  - **Tipo**: Array;
  - **Descrizione**: contiene i riferimenti agli id dei frame scelta selezionabili dal frame.

- **backgroundimage**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: contiene il riferimento dell'immagine di sfondo frame.
- **backgroundcolor**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta il colore dello sfondo del frame.

Metodi

- **Frame**(id:integer, posX:double, posY:double, degrees:double)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto, imposta i campi id, xIndex, yIndex, rotation.
- **setPrev**(prevId: integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta il campo prev.
- **setNext**(nextId: integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta il campo next.
- **setBackgroundImage**(ref:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta il campo BackgroundImage.
- **setBackgroundColor**(newColor:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta il campo BackgroundColor.
- **isBookmark**()
  - **Accesso**: Public;

  - **Tipo di ritorno**: Bool;
  - **Descrizione**: ritorna il valore del campo bookmark.

- **setBookmark**(value: bool = 1)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta il campo bookmark.
- **addChoice**(frameId:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: aggiunge una scelta all'array choices.
- **removeChoice**(frameId:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: rimuove una scelta dall'array choices.
- **getBackgroundImage**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: String;
  - **Descrizione**: restituisce il valore di backgroundImage.
- **getBackgroundColor**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: String;
  - **Descrizione**: restituisce il valore di backgroundColor.
- **getPrev**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Integer;
  - **Descrizione**: restituisce il valore di prev.
- **getNext**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Integer;
  - **Descrizione**: restituisce il valore di next.
- **getCoices**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Array;
  - **Descrizione**: restituisce il valore di choiches.

5.1.4 Classe SVG

Funzione
Classe concreta, i suoi elementi rappresentano un oggetto di tipo SVG.
Scope
Model::SlideShow::SlideShowElements::SVG.
Utilizzo
Il costruttore viene invocato da Inserter::insertSVG().
Attributi

- **color**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta il colore dell'oggetto.
- **shape**
  - **Accesso**: Private;
  - **Tipo**: Array;
  - **Descrizione**: rappresenta le coordinate della forma dell'oggetto.

Metodi

- **SVG**(id:integer, posX:double, posY:double, degrees:double, color:string, shape:array)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto, imposta i campi id, xIndex, yIndex, rotation, color, shape.
- **setColor**(newColor:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta il campo color dell'oggetto.
- **setShape**(newShape:array)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta il campo shape dell'oggetto.
- **getColor**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: String;

  - **Descrizione**: restituisce il valore di color.

- **getShape**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Array;
  - **Descrizione**: restituisce il valore di shape.

5.1.5 Classe Audio

Funzione
Classe concreta, i suoi elementi rappresentano un oggetto di tipo immagina.
Scope
Model::SlideShow::SlideShowElements::Audio.
Utilizzo
Il costruttore viene invocato da Inserter::insertAudio().
Attributi

- **url**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta il percorso dell'oggetto.

Metodi

- **Audio**(id:integer, posX:double, posY:double, degrees:double, ref:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto, imposta i campi id, xIndex, yIndex, rotation e url.
- **getUrl**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: String;
  - **Descrizione**: restituisce il valore di url.

5.1.6 Classe Video

Funzione
Classe concreta, i suoi elementi rappresentano un oggetto di tipo video.
Scope
Model::SlideShow::SlideShowElements::Video.
Utilizzo
Il costruttore viene invocato da Inserter::insertVideo().
Attributi

- **url**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta il percorso dell'oggetto.

Metodi

- **Video(id:integer, posX:double, posY:double, degrees:double, ref:string)**
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto, imposta i campi id, xIndex, yIndex, rotation e url.
- **getUrl**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: String;
  - **Descrizione**: restituisce il valore di url.

5.1.7 Classe Background

Funzione
Classe concreta, i suoi elementi rappresentano lo sfondo.
Scope
Model::SlideShow::SlideShowElements::Background.
Utilizzo
Il costruttore viene invocato da Inserter::insertBackground().
Attributi

- **url**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta il riferimento dell'immagine dello sfondo.
- **color**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta il colore dello sfondo.

Metodi

- **Background**(id:integer, color:string, ref:string="undefined")
  - **Accesso**: Public;

  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto, imposta i campi id, xIndex, yIndex, rotation, color e url.

- **setColor**(newColor:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta il campo color dell'oggetto.

- **setUrl**(ref:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta il campo url dell'oggetto.

- **getUrl**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: String;
  - **Descrizione**: restituisce il valore di url.

- **getColor**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: String;
  - **Descrizione**: restituisce il valore di color.

5.2 Classe InsertEditRemove

5.2.1 Classe Inserter

Funzione
Classe statica in cui vengono implementati gli algoritmi di inserimento di elementi nella presentazione.
Scope
Model::SlideShow::SlideShowActions::InsertEditRemove.
Utilizzo
Viene utilizzata dalla classe command per eseguire i comandi di inserimento.
Attributi

- **Presentazione**
  - **Accesso**: Private;
  - **Descrizione**: oggetto json che contiene gli oggetti delle classi che rappresentano gli elementi della presentazione.

- **id =0**
  - **Accesso**: Private;
  - **Descrizione**: attributo statico, indica gli id univoci degli oggetti generati.

Metodi

- **insertText**(posX:double, posY:double, degrees:double)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Integer;
  - **Descrizione**: costruisce un oggetto newText di tipo Text (id:integer, posX:double, posY:double, degrees:double) invocandone il costruttore passando come parametri id:integer, posX, posY e degrees. Inserisce l'oggetto così costruito nell'oggetto Presentazione, copia id in un intero tempid, esegue id++ e restituisce tempid.
- **insertText**(oldText:Text)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: inserisce l'oggetto passato per parametro nell'oggetto Presentazione.
- **setUrl**(ref:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: imposta il campo url dell'oggetto.
- **insertFrame**(posX:double, posY:double, degrees:double)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Integer;
  - **Descrizione**: costruisce un oggetto di tipo Frame(id:integer, posX:double, posY:double, degrees:double) invocandone il costruttore passando come parametri id:integer, posX, posY e degrees. Inserisce l'oggetto così costruito nell'oggetto Presentazione, copia id in un intero tempid, esegue id++ e restituisce tempid.
- **insertImage**(posX, posY, rotation, ref)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Integer;
  - **Descrizione**: costruisce un oggetto di tipo Image(id:integer, posX:double, posY:double, degrees:double, ref:string) invocandone il costruttore passando come parametri id:integer, posX, posY, degrees e ref. Copia id in un intero tempid, esegue id++ e restituisce tempid.
- **insertImage**(oldImage:Image)

  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: inserisce l'oggetto passato per parametro nell'oggetto Presentazione.

- **insertSVG**(posX:double, posY:double, rotation:double, shape:string, color:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Integer;
  - **Descrizione**: invoca il costruttore di un oggetto SVG(id:integer, posX:double, posY:double, degrees:double, shape:string, color:string). Inserisce l'oggetto così costruito nell'oggetto Presentazione, copia id in un intero tempid, esegue id++ e restituisce tempid.

- **insertSVG**(oldSVG:SVG)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: inserisce l'oggetto passato per parametro nell'oggetto Presentazione.

- **insertAudio**(posX:double, posY:double, degrees:double, ref:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Integer;
  - **Descrizione**: invoca il costruttore di un oggetto Audio(id:integer, posX:double, posY:double, degrees:double, ref:string). Inserisce l'oggetto così costruito nell'oggetto Presentazione, copia id in un intero tempid, esegue id++ e restituisce tempid.

- **insertAudio**(oldAudio:Audio)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: inserisce l'oggetto passato per parametro nell'oggetto Presentazione.

- **insertVideo**(posX:double, posY:double, degrees:double, ref:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Integer;
  - **Descrizione**: invoca il costruttore di un oggetto Video(id:integer, posX:double, posY:double, degrees:double, ref:string). Inserisce l'oggetto così costruito nell'oggetto Presentazione, copia id in un intero tempid, esegue id++ e restituisce tempid.

- **insertVideo**(oldVideo:Video)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: inserisce l'oggetto passato per parametro nell'oggetto Presentazione.

- **insertBackground**(ref:string, color:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Integer;
  - **Descrizione**: invoca il costruttore di un oggetto Background(id:integer, ref:string, color:string). Inserisce l'oggetto così costruito nell'oggetto Presentazione, copia id in un intero tempid, esegue id++ e restituisce tempid.
- **insertBackground**(oldBackground:Background)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: inserisce l'oggetto passato per parametro nell'oggetto Presentazione.
- **removeText**(id:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Text;
  - **Descrizione**: copia l'oggetto con il campo dati id corrispondente e lo rimuove dall'oggetto Presentazione. Restituisce l'oggetto copiato.
- **removeFrame**(id:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Frame;
  - **Descrizione**: copia l'oggetto con il campo dati id corrispondente, accede al suo campo prev e ne identifica i predecessore, pone prev.next=next. Rimuove l'oggetto dall'oggetto Presentazione e restituisce l'oggetto copiato.
- **removeImage**(id:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Image;
  - **Descrizione**: copia l'oggetto con il campo dati id corrispondente e lo rimuove dall'oggetto Presentazione. Restituisce l'oggetto copiato.
- **removeSVG**(id:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: SVG;
  - **Descrizione**: copia l'oggetto con il campo dati id corrispondente e lo rimuove dall'oggetto Presentazione. Restituisce l'oggetto copiato.
- **removeAudio**(id:integer)
  - **Accesso**: Public;

  - **Tipo di ritorno**: Audio;
  - **Descrizione**: copia l'oggetto con il campo dati id corrispondente e lo rimuove dall'oggetto Presentazione. Restituisce l'oggetto copiato.

- **removeVideo**(id:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Video;
  - **Descrizione**: copia l'oggetto con il campo dati id corrispondente e lo rimuove dall'oggetto Presentazione. Restituisce l'oggetto copiato.

- **removeBackground**(id:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Background;
  - **Descrizione**: copia l'oggetto con il campo dati id corrispondente e lo rimuove dall'oggetto Presentazione. Restituisce l'oggetto copiato.

- **editPosition**(id:integer, tipo:string, posX, posY)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Array[2] di double; [[[[[[[[[[[[E' CORRETTO?]]]]]]]]]]]]
  - **Descrizione**: scorre il campo dati che contiene gli oggetti di tipo SlideShowElements in Presentazione per trovare l'oggetto con il campo id corrispondente, crea una coppia oldPosition di double settati con il valore di xIndex e di yIndex dell'oggetto trovato e imposta xIndex con il valore di posX e yIndex con il valore di posY. Restituisce oldPosition.

- **editRotation**(id:integer, tipo:string, degrees)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Double;
  - **Descrizione**: scorre il campo dati che contiene gli oggetti di tipo SlideShowElements in Presentazione per trovare l'oggetto con il campo id corrispondente, crea un double oldRotation settato con il valore di rotation dell'oggetto trovato e imposta rotation con il valore di degrees. Restituisce oldRotation.

- **editSize**(id:integer, tipo:string, newHeight:double, newWidth:double)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Array[2] di double; [[[[[[[[[[[[E' CORRETTO?]]]]]]]]]]]];
  - **Descrizione**: scorre il campo dati che contiene gli oggetti di tipo tipo in Presentazione per trovare l'oggetto con il campo id corrispondente, crea una coppia oldSize di double settati con il valore di height e di width dell'oggetto trovato e imposta height con il valore di newHeight e width con il valore di newWidth. Restituisce oldSize.

- **editBackground**(id:integer, tipo:string, newColor:string, newRef:string=”undefined”)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Array[2] di double; [[[[[[[[[[[[E’ CORRETTO?]]]]]]]]]]]];
  - **Descrizione**: scorre il campo dati che contiene gli oggetti di tipo SlideShowElements in Presentazione per trovare l’oggetto con il campo id corrispondente. Se lo trova crea una coppia oldBackground di string settati con il valore di color e di ref dell’oggetto trovato e imposta color con il valore di newColor e ref con il valore di newRef. Restituisce oldBackground.
- **editColor**(id:integer, tipo:string, newColor:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: String;
  - **Descrizione**: scorre il campo dati che contiene gli oggetti di tipo SlideShowElements in Presentazione per trovare l’oggetto con il campo id corrispondente. Se lo trova copia il campo color in una string oldColor e imposta color con il valore di newColor. Restituisce oldColor.
- **editShape**(id:integer, tipo:string, newShape:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Array di integer;
  - **Descrizione**: scorre il campo dati che contiene gli oggetti di tipo SlideShowelements in Presentazione per trovare l’oggetto con il campo id corrispondente. Se lo trova copia il campo shape in un array oldShape e imposta shape con il valore di newShape. Restituisce oldShape.

5.3 Classe Command

5.3.1 Classe Invoker

Funzione
Classe, componente invoker del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::Invoker.
Utilizzo
Viene utilizzata per eseguire i comandi e memorizzarli all'interno di stack dedicate all'implementazione delle funzionalità di annulla e ripristina.
Attributi

- **undoStack**
  - **Accesso**: Private;
  - **Tipo**: Array;
  - **Descrizione**: contiene l'elenco dei comandi eseguiti e annullabili.
- **redoStack**
  - **Accesso**: Private;
  - **Tipo**: Array;
  - **Descrizione**: contiene l'elenco dei comandi annullati e ripristinabili.

Metodi

- **execute**(AbstractCommand)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo doAction() del comando ricevuto e invoca undoStack.push(AbstractCommand) e redoStack.clear().
- **undo**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo undoAction() dell'ultimo comando in undoStack() e invoca command=undoStack.pop() e redoStack.push(command).
- **redo**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo doAction() dell'ultimo comando in redoStack() e invoca command=redoStack.pop() e undoStack.push(command).

- **getUndoStack**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Boolean;
  - **Descrizione**: ritorna true se undoStack non è vuoto, false altrimenti.
- **getRedoStack**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Boolean;
  - **Descrizione**: ritorna true se redoStack non è vuoto, false altrimenti.

5.3.2 Classe AbstractCommand

Funzione
Classe concreta, i suoi elementi rappresentano un oggetto di tipo testo.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand.
Utilizzo
È classe base per i comandi di modifica, inserimento ed eliminazione degli elementi della presentazione.
Attributi

- **id**
  - **Accesso**: Private;
  - **Tipo**: Integer;
  - **Descrizione**: indica il codice identificativo dell'oggetto su cui viene eseguito il comando.
- **executed**
  - **Accesso**: Private;
  - **Tipo**: Boolean;
  - **Descrizione**: è settata a false di default, indica se il comando è stato eseguito.

Metodi

- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void; [[[[[[[[[CORRETTO??]]]]]]]]]
  - **Descrizione**: metodo virtuale implementato dalle sottoclassi. Svolge le operazioni a cui è dedicato il comando.
- **undoAction**()

- **Accesso**: Public;
- **Tipo di ritorno**: Void;[[[[[[[[[[CORRETTO??]]]]]]]]]]
- **Descrizione**: metodo virtuale implementato dalle sottoclassi. Annulla le operazioni eseguite dal comando.

Ereditata da:

- ConcreteTextInsertCommand (§5.3.2.1);
- ConcreteFrameInsertCommand (§5.3.2.2);
- ConcreteImageInsertCommand (§5.3.2.3);
- ConcreteSVGInsertCommand (§5.3.2.4);
- ConcreteAudioInsertCommand (§5.3.2.5);
- ConcreteVideoInsertCommand (§5.3.2.6);
- ConcreteBackgroundInsertCommand (§5.3.2.7);
- ConcreteTextRemoveCommand (§5.3.2.8);
- ConcreteFrameRemoveCommand (§5.3.2.9);
- ConcreteImageRemoveCommand (§5.3.2.10);
- ConcreteSVGRemoveCommand (§5.3.2.11);
- ConcreteAudioRemoveCommand (§5.3.2.12);
- ConcreteVideoRemoveCommand (§5.3.2.13);
- ConcreteBackgroundRemoveCommand (§5.3.2.14);
- ConcreteEditSizeCommand (§5.3.2.17);
- ConcreteEditPositionCommand (§5.3.2.15);
- ConcreteEditRotationCommand (§5.3.2.16);
- ConcreteEditColorCommand (§5.3.2.19);
- ConcreteEditBackgroundCommand (§5.3.2.18);
- ConcreteEditFontCommand (§5.3.2.20).

5.3.2.1 Classe ConcreteTextInsertCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteTextInsertCommand.
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve le coordinate di inserimento di un testo nella presentazione e invoca il metodo di Inserter insertText() passandogliele.
Attributi

- **posX**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle x in cui si deve inserire il testo.
- **posY**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle y in cui si deve inserire il testo.
- **rotation**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta i gradi di rotazione che deve avere l'oggetto inserito.

Metodi

- **ConcreteTextInsertCommand**(posX: double, posY: double, rotation: double)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteTextInsertCommand e setta posX, posY e rotation.
- **doAction**()
  - **Accesso**: Public;
  - **Tipo**: Void;
  - **Descrizione**: invoca il metodo di Inserter insertText(posX, posY, degrees) passando come parametri i campi dati posX, posY e rotation. insertText restituisce un int che rappresenta l'id dell'oggetto. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.
- **undoAction**()

- **Accesso**: Public;
- **Tipo**: Void;
- **Descrizione**: invoca il metodo di Editor removeText(id) passando come parametro il campo id. Invoca il metodo remove(id) di EditController.

5.3.2.2 Classe ConcreteFrameInsertCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteFrameInsertCommand.
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve le coordinate di inserimento di un testo nella presentazione e invoca il metodo di Inserter insertFrame() passandogliele.
Attributi

- **posX**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle x in cui si deve inserire il frame.
- **posY**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle y in cui si deve inserire il frame.
- **rotation**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta i gradi di rotazione che deve avere l'oggetto inserito.

Metodi

- **ConcreteFrameInsertCommand**(posX: double, posY: double, rotation: double)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteFrameInsertCommand e setta posX, posY e rotation.
- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;

  - **Descrizione**: invoca il metodo di Inserter insertFrame(posX, posY, rotation) passando come parametri i campi dati posX, posY e rotation. insertFrame restituisce un int che rappresenta l'id dell'oggetto. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.

- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo di Editor removeFrame(id) passando come parametro il campo id. Invoca il metodo remove(id) di EditController.

5.3.2.3 Classe ConcreteImageInsertCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteImageInsertCommand.
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve le coordinate di inserimento di un'immagine nella presentazione e invoca il metodo di Inserter insertImage() passandogliele.
Attributi

- **posX**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle x in cui si deve inserire il frame.
- **posY**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle y in cui si deve inserire il frame.
- **rotation**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta i gradi di rotazione che deve avere l'oggetto inserito.
- **ref**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta l'url dell'oggetto inserito.

Metodi

- **ConcreteImageInsertCommand**(posX: double, posY: double, rotation: double, ref: string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteImageInsertCommand e setta posX, posY, rotation e ref.
- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo di Inserter insertImage(posX, posY, rotation, ref) passando come parametri i campi dati posX, posY, rotation e ref. insertImage restituisce un int che rappresenta l'id dell'oggetto. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.
- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo di Editor removeImage(id) passando come parametro il campo id. Invoca il metodo remove(id) di EditController.

5.3.2.4 Classe ConcreteSVGInsertCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteSVGInsertCommand.
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve le coordinate di inserimento di un SVG nella presentazione e invoca il metodo di Inserter insertSVG() passandogliele.
Attributi

- **posX**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle x in cui si deve inserire il frame.
- **posY**
  - **Accesso**: Private;
  - **Tipo**: Double;

  - **Descrizione**: rappresenta la posizione sull'asse delle y in cui si deve inserire il frame.

- **rotation**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta i gradi di rotazione che deve avere l'oggetto inserito.

- **shape**
  - **Accesso**: Private;
  - **Tipo**: Array;
  - **Descrizione**: rappresenta la forma dell'oggetto inserito.

- **color**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta il colore dell'oggetto inserito.

Metodi

- **ConcreteSVGInsertCommand**(posX: double, posY: double, rotation: double, shape:array, color:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteSVGInsertCommand e setta posX, posY, rotation, shape, color.

- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo di Inserter insertSVG(posX, posY, shape, color) passando come parametri i campi dati posX, posY, rotation, shape e color. insertSVG restituisce un int che rappresenta l'id dell'oggetto. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.

- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo di Editor removeSVG(id) passando come parametro il campo id. Invoca il metodo remove(id) di EditController.

5.3.2.5 Classe ConcreteAudioInsertCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteAudioInsertCommand.
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve le coordinate di inserimento di un audio nella presentazione e invoca il metodo di Inserter insertAudio() passandogliele.
Attributi

- **posX**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle x in cui si deve inserire il frame.
- **posY**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle y in cui si deve inserire il frame.
- **rotation**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta i gradi di rotazione che deve avere l'oggetto inserito.
- **ref**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta l'url dell'oggetto inserito.

Metodi

- **ConcreteAudioInsertCommand**(posX: double, posY: double, rotation: double, ref: string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteAudioInsertCommand e setta posX, posY, rotation, ref.
- **doAction**()
  - **Accesso**: Public;

  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo di Inserter insertAudio(posX, posY, rotation, ref) passando come parametri i campi dati posX, posY, rotation e ref. insertAudio restituisce un int che rappresenta l'id dell'oggetto. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.

- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo di Editor removeAudio(id) passando come parametro il campo id. Invoca il metodo remove(id) di EditController.

5.3.2.6 Classe ConcreteVideoInsertCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteVideoInsertCommand.
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve le coordinate di inserimento di un video nella presentazione e invoca il metodo di Inserter insertVideo() passandogliele..
Attributi

- **posX**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle x in cui si deve inserire il frame.
- **posY**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta la posizione sull'asse delle y in cui si deve inserire il frame.
- **rotation**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: rappresenta i gradi di rotazione che deve avere l'oggetto inserito.
- **ref**
  - **Accesso**: Private;
  - **Tipo**: String;

- **Descrizione**: rappresenta l'url dell'oggetto inserito.

Metodi

- **ConcreteVideoInsertCommand**(posX: double, posY: double, rotation: double, ref: string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteVideoInsertCommand e setta posX, posY, rotation, ref.
- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: ivoca il metodo di Inserter insertVideo(posX, posY, rotation, ref) passando come parametri i campi dati posX, posY, rotation e ref. insertVideo restituisce un int che rappresenta l'id dell'oggetto. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.
- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo di Editor removeVideo(id) passando come parametro il campo id. Invoca il metodo remove(id) di EditController.

5.3.2.7 Classe ConcreteBackgroundInsertCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteBackgroundInsertComm
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve i parametri di inserimento di uno sfondo nella presentazione e invoca il metodo di Inserter insertBackground() passandogliele.
Attributi

- **ref**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta l'url dell'oggetto inserito.
- **color**

  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: rappresenta il colore dello sfondo inserito.

- **oldBackground**
  - **Accesso**: Private;
  - **Tipo**: SlideShowElements::Background;
  - **Descrizione**: rappresenta il vecchio sfondo della presentazione.

Metodi

- **ConcreteBackgroundInsertCommand**(ref: string, color:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteVideoInsertCommand e setta ref e color.
- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo di Inserter insertBackground (ref, color) passando come parametri i campi dati ref e color. insertBackground restituisce un int che rappresenta l'id dell'oggetto o eventualmente un oggetto di tipo Background a cui ConcreteBackgroundInsertCommand istanzia oldBackground e al cui id inizializza il campo id. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.
- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo di Remover removeBackground(id) passando come parametro il campo id. Se il campo oldBackground è inizializzato invoca il metodo Inserter::insertBackground(SlideShowElement::Background) e invoca il metodo update(id) di EditController, altrimenti ne invoca il metodo remove(id).

5.3.2.8 Classe ConcreteTextRemoveCommand

Funzione

Classe concreta, è interfaccia del Design Pattern Command..

Scope

Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteTextRemoveCommand.

Utilizzo

Viene costruito da Premi::Controller::EditController, riceve l'id dell'elemento testo da rimuovere dalla presentazione e invoca il metodo di Remover removeText(id).

Attributi

- **text**
  - **Accesso**: Private;
  - **Tipo**: SlideShowElements::Text;
  - **Descrizione**: è una copia dell'elemento testo rimosso.

Metodi

- **ConcreteTextRemoveCommand**(id:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteTextRemoveCommand e setta l'attributo id.
- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Remover::removeText(id) passando come parametro l'id dell'elemento. removeText restituisce un oggetto di tipo SlideShowElements::Text. Se executed è settato a false lo setta come true, altrimenti invoca il metodo EditController::update(id).
- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Inserter::insertText(Text) passando come parametro text. Invoca il metodo update(id) di EditController.

5.3.2.9 Classe ConcreteFrameRemoveCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteFrameRemoveComman
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve l'id del frame da rimuovere dalla presentazione e invoca il metodo Remover::removeFrame(id).
Attributi

- **removedFrame**
  - **Accesso**: Private;
  - **Tipo**: SlideShowElements::Frame;
  - **Descrizione**: è una copia dell'oggetto rimosso.

Metodi

- **ConcreteFrameRemoveCommand**(id:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteFrameRemoveCommand e setta i campi dati id.
- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Remover::removeFrame(id) passando come parametro il campi dati id. removeFrame restituisce una copia dell'oggetto rimosso che verrà settata come campo dati removedFrame. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.
- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Inserter::insertFrame(Frame) passando come parametro il campo removedFrame. Invoca il metodo update(id) di EditController.

5.3.2.10 Classe ConcreteImageRemoveCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteImageRemoveComman
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve l'id dell'immagine da rimuovere dalla presentazione e invoca il metodo Remover::removeImage(id).
Attributi

- **removedImage**
  - **Accesso**: Private;
  - **Tipo**: SlideShowElements::Image;
  - **Descrizione**: è una copia dell'oggetto rimosso

Metodi

- **ConcreteImageRemoveCommand**(id: integer)
  - **Accesso**: Public;

  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteImageRemoveCommand e setta il campo id.

- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Remover::removeImage(id) passando come parametro l'id dell'oggetto da rimuovere. removeImage restituisce un oggetto di tipo SlideShowElements::Image cui sarà settato il campo removedImage. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.

- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Inserter::insertImage(Image) passando come parametro il campo removedImage. Invoca il metodo update(id) di EditController.

5.3.2.11 Classe ConcreteSVGRemoveCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteSVGRemoveCommand.
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve l'id dell'elemento SVG da rimuovere dalla presentazione e invoca il metodo Remover::removeSVG().
Attributi

- **removedSVG**
  - **Accesso**: Private;
  - **Tipo**: SlideShowElements::SVG;
  - **Descrizione**: è una copia dell'oggetto che rappresenta l'elemento rimosso.

Metodi

- **ConcreteSVGRemoveCommand**(id:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteSVGRemoveCommand e setta il campo dati id.

- **doAction**()

  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Remover::removeSVG(id) passando come parametro il campo dati id. removeSVG restituisce un elemento di tipo SlideShowElements::SVG che rappresenta l'elemento rimosso e a cui viene inizializzato il campo removedSVG. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.

- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo di Inserter::insertSVG(SVG) passando come parametro il campo removedSVG. Invoca il metodo update(id) di EditController.

5.3.2.12 Classe ConcreteAudioRemoveCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteAudioRemoveCommand
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve l'id dell'elemento audio da rimuovere dalla presentazione e invoca il metodo Remover::removeAudio(id).
Attributi

- **removedAudio**
  - **Accesso**: Private;
  - **Tipo**: SlideShowElements::Audio;
  - **Descrizione**: è una copia dell'oggetto che rappresenta l'elemento rimosso.

Metodi

- **ConcreteAudioRemoveCommand**(id:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteAudioRemoveCommand e setta il campo dati id.

- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;

- **Descrizione**: invoca il metodo Remover::removeAudio(id) passando come parametro il campo dati id. removeAudio restituisce una copia dell'oggetto Audio rimosso. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.

- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Inserter::insertAudio(Audio) passando come parametro il campo removedAudio. Invoca il metodo update(id) di EditController.

5.3.2.13 Classe ConcreteVideoRemoveCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteVideoRemoveCommand
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve l'id dell'elemento video da rimuovere dalla presentazione e invoca il metodo Remover::removeVideo(id).
Attributi

- **removedVideo**
  - **Accesso**: Private;
  - **Tipo**: SlideShowElements::Video;
  - **Descrizione**: è una copia dell'oggetto che rappresenta l'elemento rimosso.

Metodi

- **ConcreteVideoRemoveCommand**(id:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteVideoRemoveCommand e setta il campo dati id.

- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Remover::removeVideo(id) passando come parametro il campo dati id. removeVideo restituisce una copia dell'oggetto Video rimosso. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.

- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Inserter::insertVideo(Video) passando come parametro il campo removedVideo. Invoca il metodo update(id) di EditController.

5.3.2.14 Classe ConcreteBackgroundRemoveCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteBackgroundRemoveCon
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve l'id dell'elemento sfondo da rimuovere dalla presentazione e invoca il metodo Remover::removeBackground(id).
Attributi

- **removedBackground**
  - **Accesso**: Private;
  - **Tipo**: SlideShowElements:: Background;
  - **Descrizione**: è una copia dell'oggetto che rappresenta l'elemento rimosso.

Metodi

- **ConcreteBackgroundRemoveCommand**(id:integer)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteBackgroundRemoveCommand e setta il campo dati id.
- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Remover::removeBackground (id) passando come parametro il campo dati id. removeBackground restituisce una copia dell'oggetto Audio rimosso. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.
- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Inserter::insertBackground (Background) passando come parametro il campo removedBackground. Invoca il metodo update(id) di EditController.

5.3.2.15 Classe ConcreteEditPositionCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteEditPositionCommand.
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve l'id dell'elemento dalla presentazione e le coordinate x e y in cui deve essere spostato l'elemento, invoca il metodo Editor::editPosition(id, posX, posY).
Attributi

- **type**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: indica il tipo dell'elemento da modificare.
- **oldPosX**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: indica la vecchia posizione sull'asse x dell'elemento.
- **oldPosY**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: indica la vecchia posizione sull'asse y dell'elemento.
- **newPosX**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: indica la nuova posizione sull'asse x dell'elemento.
- **newPosY**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: indica la nuova posizione sull'asse y dell'elemento.

Metodi

- **ConcreteEditPositionCommand**(id:integer, tipo:string, posX:double, posY:double)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;

  - **Descrizione**: costruisce l'oggetto ConcreteEditPositionCommand e setta il campo dati id, il campo dati type il campo dati newPosX e il campo dati newPosY con i parametri ricevuti.

- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Editor::editPosition(id, tipo, posX, posY) passando come parametri i campi dati id, type, new posX e newPosY. editPosition ritorna una coppia di double a cui ConcreteEditPositionCommand inizializza oldPosX e oldPosY. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.

- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Editor::editPosition(id, type, posX, posY) passando come parametro i campi dati id, type, oldPosX, oldPosY. Invoca il metodo update(id) di EditController.

5.3.2.16 Classe ConcreteEditRotationCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteEditRotationCommand
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve l'id dell'elemento dalla presentazione e il grado a cui deve essere ruotato l'elemento, invoca il metodo Editor::editRotation(id, tipo, degrees).
Attributi

- **type**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: indica il tipo dell'elemento da modificare.

- **oldDegrees**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: indica i vecchi gradi di rotazione dell'elemento.

- **newDegrees**

- **Accesso**: Private;
- **Tipo**: Double;
- **Descrizione**: indica i nuovi gradi di rotazione dell'elemento.

Metodi

- **ConcreteEditRotationCommand**(id:integer, tipo:string, degrees:double)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteEditRotationCommand e setta il campo dati id, il campo dati type e il campo dati newDegrees con i parametri ricevuti.
- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Editor::editRotation(id, tipo, degrees) passando come parametri i campi dati id, type e newDegrees. editRotation ritorna un double cui ConcreteEditRotationCommand inizializza oldDegrees. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.
- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Editor::editRotation(id, tipo, degrees) passando come parametro i campi dati id, tipo, oldRotation. Invoca il metodo update(id) di EditController.

5.3.2.17 Classe ConcreteEditSizeCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteEditSizeCommand.
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve l'id dell'elemento dalla presentazione e le dimensioni a cui deve essere ridimensionato l'elemento, invoca il metodo Editor::editSize(id, tipo, height, width).
Attributi

- **type**
  - **Accesso**: Private;
  - **Tipo**: String;

  - **Descrizione**: indica il tipo dell'elemento da modificare.

- **oldHeight**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: indica la vecchia altezza dell'elemento.

- **oldWidth**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: indica la vecchia larghezza dell'elemento.

- **newHeight**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: indica la nuova altezza dell'elemento.

- **newWidth**
  - **Accesso**: Private;
  - **Tipo**: Double;
  - **Descrizione**: indica la nuova larghezza dell'elemento.

Metodi

- **ConcreteEditSizeCommand**(id:integer, tipo:string, height:double, width:double)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteEditSizeCommand e setta il campo dati id, il campo dati type e i campi dati newHeight e newWidth con i parametri ricevuti.

- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Editor::editSize(id, tipo, height, width) passando come parametri i campi dati id, type, newHeight e newWidth. editSize ritorna una coppia di double a cui ConcreteEditSizeCommand inizializza oldHeight e oldWidth. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.

- **undoAction**()

- **Accesso**: Public;
- **Tipo di ritorno**: Void;
- **Descrizione**: invoca il metodo Editor::editSize(id, tipo, height, width) passando come parametro i campi dati id, tipo, oldHeight, oldWidth. Invoca il metodo update(id) di EditController.

5.3.2.18 Classe ConcreteEditBackgroundCommand

Funzione

Classe concreta, è interfaccia del Design Pattern Command.

Scope

Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteEditBackgroundComma

Utilizzo

Viene costruito da Premi::Controller::EditController, riceve l'id dell'elemento dalla presentazione e lo sfondo da applicargli, invoca il metodo Editor::editBackground(id, tipo, newRef, newColor).

Attributi

- **type**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: indica il tipo dell'elemento da modificare.
- **oldRef**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: indica il vecchio url dell'immagine di sfondo dell'elemento.
- **oldColor**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: indica il vecchio colore di sfondo dell'elemento.
- **newRef**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: indica il nuovo url dell'immagine di sfondo dell'elemento.
- **newColor**
  - **Accesso**: Private;
  - **Tipo**: String;

- **Descrizione**: indica il nuovo colore di sfondo dell'elemento.

Metodi

- **ConcreteEditBackgroundCommand**(id:integer, tipo:string, url:string, color:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteEditBackgroundCommand e setta il campo dati id, il campo dati type e i campi dati newUrl e newColor con i parametri ricevuti.
- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Editor::editBackground(id, tipo, newRef, newColor) passando come parametri i campi dati id, type, newRef e newColor. editBackground ritorna una ConcreteEditBackgroundCommand inizializza oldRef e oldColor. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.
- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Editor::editBackground(id, tipo, url, color) passando come parametro i campi dati id, tipo, oldUrl, oldColor. Invoca il metodo update(id) di EditController.

5.3.2.19 Classe ConcreteEditColorCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteEditColorCommand.
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve l'id dell'elemento dalla presentazione e il colore da applicargli, invoca il metodo Editor::editColor(id, tipo, color).
Attributi

- **type**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: indica il tipo dell'elemento da modificare.
- **oldColor**

  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: indica il vecchio colore di sfondo dell'elemento.

- **newColor**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: indica il nuovo colore di sfondo dell'elemento.

Metodi

- **ConcreteEditColorCommand**(id:integer, tipo:string, color:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteEditColorCommand e setta il campo dati id, il campo dati type e il campo dati newColor con i parametri ricevuti.
- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Editor::editColor(id, tipo, color) passando come parametri i campi dati id e newColor. editColor ritorna una variabile di tipo string a cui ConcreteEditColorCommand inizializza oldColor. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.
- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Editor::editColor(id, tipo, color) passando come parametro i campi dati id,tipo, oldColor. Invoca il metodo update(id) di EditController.

5.3.2.20 Classe ConcreteEditFontCommand

Funzione
Classe concreta, è interfaccia del Design Pattern Command.
Scope
Model::SlideShow::SlideShowActions::Command::AbstractCommand::ConcreteEditFontCommand.
Utilizzo
Viene costruito da Premi::Controller::EditController, riceve l'id dell'elemento dalla presentazione e lo sfondo da applicargli, invoca il metodo Editor::editFont(id, tipo, font).
Attributi

- **type**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: indica il tipo dell'elemento da modificare.
- **oldFont**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: indica il vecchio font dell'elemento.
- **newFont**
  - **Accesso**: Private;
  - **Tipo**: String;
  - **Descrizione**: indica il nuovo font dell'elemento.

Metodi

- **ConcreteEditFontCommand**(id:integer, tipo:string, font:string)
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: costruisce l'oggetto ConcreteEditFontCommand e setta il campo dati id, il campo dati type e il campo dati newFont con i parametri ricevuti.
- **doAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Editor::editFont (id, tipo, font) passando come parametri i campi dati id, type, newFont. editFont ritorna una variabile di tipo stringa a cui ConcreteEditFontCommand inizializza oldFont. Se executed è settato a false lo setta come true, altrimenti invoca il metodo update(id) di EditController.
- **undoAction**()
  - **Accesso**: Public;
  - **Tipo di ritorno**: Void;
  - **Descrizione**: invoca il metodo Editor::editFont(id, tipo, font) passando come parametro i campi dati id, tipo, oldFont. Invoca il metodo update(id) di EditController.

5.4 serverRelation

5.4.1 Loader

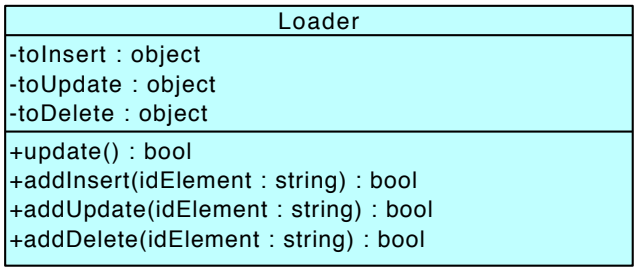

Fig 2: Diagrama classe Model::serverRelation::loader::Loader

Attributi:

- **- toInsert**
  - **Accesso:** private
  - **Tipo:** object
  - **Descrizione:** array di identificativi di elementi inseriti in locale dall'ultima sincronizzazione verso MongoDB
- **- toUpdate**
  - **Accesso:** private
  - **Tipo:** object
  - **Descrizione:** array di identificativi di elementi modificati in locale dall'ultima sincronizzazione verso MongoDB
- **- toDelete**
  - **Accesso:** private
  - **Tipo:** object
  - **Descrizione:** array di identificativi di elementi eliminati in locale dall'ultima sincronizzazione verso MongoDB

Metodi:

- **update() : bool**
  - **Accessibilità:** public
  - **Descrizione:** scorre gli array toInsert, toUpdate, toDelete e chiama le funzioni in mongoRelation per la sincronizzazione della presentazione sul database MongoDB
  - **Tipo di ritorno:** bool
- **addInsert(idElement : string) : bool**

  - **Accessibilità:** public
  - **Descrizione:** inserisce la stringa parametro in toInsert
  - **Tipo di ritorno:** bool

- **addUpdate(idElement : string) : bool**
  - **Accessibilità:** public
  - **Descrizione:** inserisce la stringa parametro in toUpdate se non è presente in toInsert
  - **Tipo di ritorno:** bool

- **addDelete(idElement : string) : bool**
  - **Accessibilità:** public
  - **Descrizione:** inserisce la stringa parametro in toDelete se non è presente in toInsert, altrimenti non inserisce nulla in toDelete e cancella l'identificativo da toInsert, sempre cancella da toUpdate
  - **Tipo di ritorno:** bool

5.5 serverRelation

5.5.1 Registration

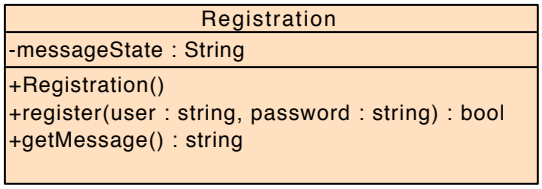

Fig 3: Diagramma classe Model::serverRelation::accessControll::Registration

Attributi:

- **-messageState**
  - **Accesso:** private
  - **Tipo:** string
  - **Descrizione:** messaggio ritornato dall'ultima chiamata di autenticazione al server

Metodi:

- **register(user : string, password : string) : bool**
  - **Accessibilità:** public
  - **Descrizione:** chiama il servizio /account/register POST con passando i parametri attuali, scrive in messageState lo stato ritornato dalla chiamata a nodeApi, ritorna true solo se la chiamata va a buon fine
  - **Tipo di ritorno:** bool
- **getMessage() : string**
  - **Accessibilità:** public
  - **Descrizione:** ritorna il contenuto di messageState
  - **Tipo di ritorno:** string

5.5.2 Authentication

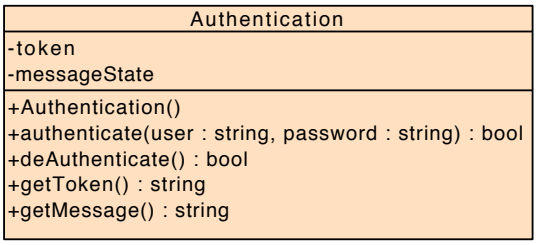

Fig 4: Diagramma classe Model::serverRelation::accessControll::Authentication

Attributi:

- **-messageState**
  - **Accesso:** private
  - **Tipo:** string
  - **Descrizione:** messaggio ritornato dall'ultima chiamata di autenticazione al server
- **-token**
  - **Accesso:** private
  - **Tipo:** string
  - **Descrizione:** stringa per l?autenticazione ai servizi Server nodeJs
  - **Note:** valore ?empty? se non è ancora stato richiesto il token, è stato cancellato, oppure l? autenticazione è fallita

Metodi:

- **authenticate(user : string, password : string) : bool**
  - **Accessibilità:** public
  - **Descrizione:** chiama il servizo account/authenticate, ritorna true se la chiamata è andata a buon fine, in questo caso mette nel campo token il token ritornato dal server nodeJs
  - **Tipo di ritorno:** bool
- **deAuthenticate() : bool**
  - **Accessibilità:** public
  - **Descrizione:** de-autentica l?utente dai servizi server cancellando il contenuto del campo dati token, ritorna true se l?operazione va a buon fine
  - **Tipo di ritorno:** bool
- **getToken() : string**
  - **Accessibilità:** public
  - **Descrizione:** ritorna il contenuto del campo dati token
  - **Tipo di ritorno:** string
- **getMessage() : string**
  - **Accessibilità:** public
  - **Descrizione:** ritorna il contenuto di messageState
  - **Tipo di ritorno:** string

5.6 serverRelation

5.6.1 MongoRelation

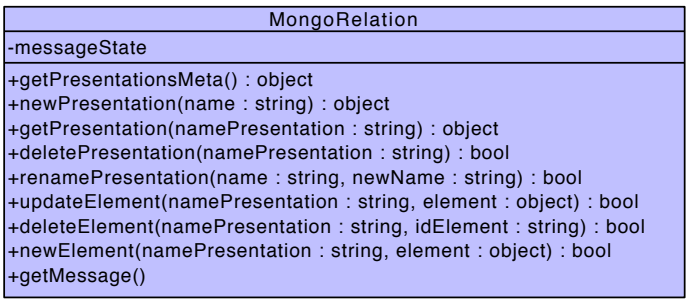

Fig 5: Diagramma classe Model::serverRelation::mongoRelation::MongoRelation

Metodi:

- **getPresentationsMeta() : object**
  - **Accessibilità:** public
  - **Descrizione:** effettua una chiamata asincrona verso il Server nodeJs al servizio "/private/api/presentations GET" ritornando l'oggetto con le informazioni sulle presentazioni create dall'utente
  - **Tipo di ritorno:** object
- **newPresentation(name : string) : object**
  - **Accessibilità:** public
  - **Descrizione:** effettua una chiamata asincrona verso il Server nodeJs al servizio "/private/api/presentations/[name] POST" per creare una nuova presentazione sul database MongoDB, una volta creata la presentazione in formato son è ritornata dal metodo
  - **Tipo di ritorno:** object
- **getPresentation(namePresentation : string) : object**
  - **Accessibilità:** public
  - **Descrizione:** effettua una chiamata asincrona verso il Server nodeJs al servizio "/private/api/presentations/[namePresentation] GET" per ricevere la presentazione [namePresentation] dell'utente
  - **Tipo di ritorno:** object
- **deletePresentation(namePresentation : string) : bool**
  - **Accessibilità:** public

  - **Descrizione:** effettua una chiamata asincrona verso il Server nodeJs al servizio "/private/api/presentations/[namePresentation] DELETE" per eliminare la presentazione [namePresentation] creata dall'utente dal database MongoDB
  - **Tipo di ritorno:** bool

- **renamePresentation(name : string, newName : string) : bool**
  - **Accessibilità:** public
  - **Descrizione:** effettua una chiamata asincrona verso il Server nodeJs al servizio "/private/api/presentations/[name]/[newName] POST" per rinominare la presentazione [name] dell'utente in [newName]
  - **Tipo di ritorno:** bool

- **updateElement(namePresentation: string, element : object) : bool**
  - **Accessibilità:** public
  - **Descrizione:** effettua una chiamata asincrona verso il Server nodeJs al servizio "/private/api/presentations/[namePresentation]/[idElement] PUT" per aggiornare l'elemento passato come parametro
  - **Tipo di ritorno:** bool

- **deleteElement(namePresentation: string, idElement : string) : bool**
  - **Accessibilità:** public
  - **Descrizione:** effettua una chiamata asincrona verso il Server nodeJs al servizio "/private/api/presentations/[namePresentation]/[idElement] DELETE" per eliminare l'elemento con identificativo [idElement] dalla presentazione nel database MongoDB
  - **Tipo di ritorno:** bool

- **newElement(namePresentation: string, element : object) : bool**
  - **Accessibilità:** public
  - **Descrizione:** effettua una chiamata asincrona verso il Server nodeJs al servizio "/private/api/presentations/[namePresentation] POST" per inserire un nuovo elemento nella presentazione dell'utente(element) nella base dati MongoDB
  - **Tipo di ritorno:** bool

5.7 serverRelation

5.7.1 Loader

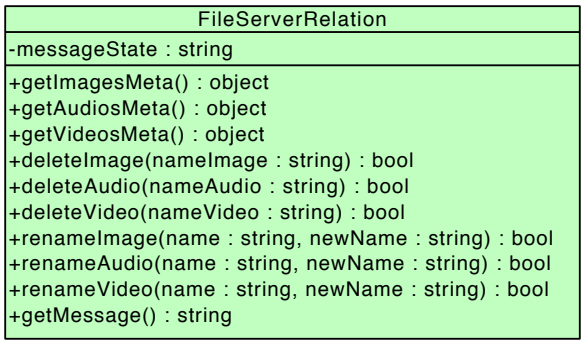

Fig 6: Diagrama classe Model::serverRelation::fileServerRelation::FileServerRelation

Attributi:

- **- messageState**
  - **Accesso:** private
  - **Tipo:** string
  - **Descrizione:** messaggio ritornato dall'ultima chiamata al Server nodeJs

Metodi:

- **getImagesMeta() : object**
  - **Accessibilità:** public
  - **Descrizione:** ritorna un oggetto contenente informazioni sulle immagini dell'utente nel server
  - **Tipo di ritorno:** object
- **getAudiosMeta() : object**
  - **Accessibilità:** public
  - **Descrizione:** ritorna un oggetto contenente informazioni sui file audio dell'utente nel server
  - **Tipo di ritorno:** object
- **getVideosMeta() : object**
  - **Accessibilità:** public
  - **Descrizione:** ritorna un oggetto contenente informazioni sui file video dell'utente nel server
  - **Tipo di ritorno:** object

- **deleteImage(nameImage : string) : bool**
  - **Accessibilità:** public
  - **Descrizione:** cancella il file immagine nameImage dallo spazio dell'utente sul server
  - **Tipo di ritorno:** bool
- **deleteAudio(nameAudio : string) : bool**
  - **Accessibilità:** public
  - **Descrizione:** cancella il file audio nameAudio dallo spazio dell'utente sul server
  - **Tipo di ritorno:** bool
- **deleteVideo(nameVideo : string) : bool**
  - **Accessibilità:** public
  - **Descrizione:** cancella il file video nameVideo dallo spazio dell'utente sul server
  - **Tipo di ritorno:** bool
- **renameImage(name : string, newName : string) : bool**
  - **Accessibilità:** public
  - **Descrizione:** rinomina il file immagine name dell'utente nel server con il nuovo nome newName
  - **Tipo di ritorno:** bool
- **renameAudio(name : string, newName : string) : bool**
  - **Accessibilità:** public
  - **Descrizione:** rinomina il file audio name dell'utente nel server con il nuovo nome newName
  - **Tipo di ritorno:** bool
- **renameVideo(name : string, newName : string) : bool**
  - **Accessibilità:** public
  - **Descrizione:** rinomina il file video name dell'utente nel server con il nuovo nome newName
  - **Tipo di ritorno:** bool

6 Package Premi::Controller

Tutti i package seguenti appartengono al package Premi, quindi per ognuno di essi lo scope sarà: Premi::[nome package].

Tipo, obiettivo e funzione del componente: contiene le classi che gestiscono i segnali e le chiamate effettuati dalla View.
Relazioni d'uso di altre componenti: comunica con il Model per la gestione del profilo e delle presentazioni.

6.1 Controller::IndexController

Funzione
Questa classe si occuperà di controllare che le credenziali di accesso siano corrette nel caso dell'autenticazione oppure di registrare un nuovo utente.
Relazioni d'uso con altri moduli
Questa classe utilizzerà le seguenti classi:

- View::Pages::Index;
- Model::MongoRelations::AccessControl::Authentication;
- Model::MongoRelations::AccessControl::Registration.

Attributi
Al momento non sono stati previsti degli attributi.
Metodi

- +index(): costruttore che dovrà inizializzare gli eventuali attributi ed inizializzare la pagina View::Pages::Index;
- +login(): metodo che richiama Model::MongoRelations::AccessControl::Authentication per effettuare l'autenticazione al sistema: se l'operazione va a buon fine, login() reindirizza alla pagina View::Pages::Home tramite il metodo goHome() altrimetti invia un segnale di errore che verrà visualizzato in View::Pages::Index;
- +subscription(): metodo che richiama Model::MongoRelations::AccessControl::Registration per registrare un nuovo utente nel sistema: se l'operazione va a buon fine, Subscription() reindirizza alla pagina View::Pages::Home tramite il metodo goHome() altrimetti segnala un errore che verrà visualizzato in View::Pages::Index;
- +goHome(): metodo che reindirizza alla pagina View::Pages::Home.

6.2 Controller::HomeController

Funzione
Questa classe si occuperà di gestire i segnali e le chiamate provenienti dalla pagina View::Pages::Home.
Relazioni d'uso con altri moduli
Questa classe utilizzerà le seguenti classi:

- View::Pages::Home;
- Model::MongoRelations::AccessControl::LoaderClass;
- Model::MongoRelations::AccessControl::Authentication;
- Model::ApacheRelations::ResourceGetter. [[[[[[[[[[[[[NOOOOOO]]]]]]]]]]]]]

Attributi

Al momento non sono stati previsti degli attributi.

Metodi

- +home(): costruttore che dovrà inizializzare gli eventuali attributi e che dovrà inizializzare la pagina View::Pages::Home in base all'id del token dell'utente attivo e riportando l'elenco delle presentazioni dell'utente richiamando Model::MongoRelations::AccessControl::LoaderClass. Nel caso in cui il token risultasse non valido o scaduto, verrà invocato il metodo goIndex() che riporterà alla pagina di autenticazione;
- +goExecute(string id): metodo che reindirizza alla pagina View::Pages::Execute passando l'id della presentazione da eseguire;
- +goIndex(): metodo che reindirizza alla pagina View::Pages::Index;
- +goProfile(): metodo che reindirizza alla pagina View::Pages::Profile per la gestione del profilo utente;
- +goEdit(string id): metodo che reindirizza alla pagina View::Pages::Edit passando l'id della presentazione da modificare;
- +deleteSlideShow(string id): metodo per l'eliminazione di una presentazione. Esso riceve l'id della presentazione dalla pagina View::Pages::Home e chiama il metodo [[[[[[[[[[METODO ELIMINAZIONE PRESENTAZIONE]]]]]]]]]] che rimuoverà la presentazione dal database e comunicherà alla pagina View::Pages::Home l'avvenuta eliminazione con conseguente rimozione dalla pagina della miniatura della presentazione in oggetto. In caso di errore esso verrà visulizzato da View::Pages::Home;
- +download(string id): metodo che riceve l'id della presentazione da View::Pages::Home e permette di poterla scaricare in locale. Il metodo richiama [[[[[[[[[METODO DOWNLOAD PRESENTAZIONE]]]]]]]]];
- +renameSlideShow(string id): metodo che riceve da View::Pages::Home l'id della presentazione e una stringa con cui rinominarlo. Il metodo richiamerà [[[[[[[[[[METODO RINOMINA PRESENTAZIONE]]]]]]]]]] che si preoccuperà di rinominarla nel database e, se l'operazione è andata a buon fine, renameSlideShow() lo comunicherà a View::Pages::Home, altrimenti segnalerà un errore;
- +execute(string id): metodo che richiama la funzione goExecute();
- +editSlideShow(): metodo che richiama la funzione goEdit(id); l'id passato sarà quello della presentazione da modificare;

- +logout(): metodo che richiama deAuthenticate() fornito da Model::MongoRelations::AccessControl:: permettendo di effettuare il logout dal sistema; dopo il logout HomeController reindirizzerà alla pagina View::Pages::Index tramite il metodo goIndex();
- +newSlideShow(): metodo che richiama [[[[[[[[[METODO MODEL CREAZIONE PRESENTAZIONE]]]]]]]]] per la creazione di una nuova presentazione del database; se la creazione è andata a buon fine richiamerà il metodo goEdit(string id), passando l'id della nuova presentazione, in modo da entrare in modalità modifica.

6.3 Controller::ProfileController

Funzione

Questa classe consentirà la modifica dei dati personali dell'utente.

Relazioni d'uso con altri moduli

Questa classe utilizzerà le seguenti classi:

- Model::ApacheRelations::FileManager; [[[[[[[[[NOOOOOOO]]]]]]]]]
- Model::MongoRelations::AccessControl::Authentication.

Attributi

Al momento non sono stati previsti degli attributi.

Metodi

- +Profile(): costruttore che dovrà inizializzare gli eventuali attibuti e la pagina View::Pages::Profile in base all'id del token dell'utente attivo e nel caso in cui nessun token risultasse attivo richiamerà il metodo goIndex(). Profile() dialogherà con Model::ApacheRelations::FileManager [[[[[[[[[NOOOOO]]]]]]]]] per recuperare tutti i file media dell'utente caricati sul server e con Model::MongoRelations::AccessControl::Authentication per ottenere i dati personali dell'utente [[[[[[[[[I DATI PERSONALI ARRIVANO DA AUTHENTICATION??]]]]]]]]];
- +changePassword(string password): metodo che dialoga con Model::MongoRelations::AccessControl: inviandogli la stringa della nuova password da sostituire con quella vecchia. In caso di errore, esso verrà mostrato tramite la pagina View::Pages::Profile;
- +uploadMedia(string percorso): metodo che invia a Model::ApacheRelations::FileManager il percorso del file media da caricare sul proprio spazio sul server, restituisce la schermata aggiornata con la nuova miniatura del file media se il caricamento è andato a buon fine, altrimenti restituisce un errore;
- deleteMedia(string id): metodo che invia l'id del file media a Model::ApacheRelations::FileManager per rimuoverlo dal database;
- +renameSlideShow(string id, string name): metodo che invia l'id del file media e una stringa a Model::ApacheRelations::FileManager per rinominarlo;
- +goHome(): metodo che reindirizza alla pagina View::Pages::Home;
- +goIndex(): metodo che reindirizza alla pagina View::Pages::Index.

6.4 View::Pages::Execution

Funzione

Questa classe si occuperà di gestire i segnali che arrivano da View::Pages::Execution.

Relazioni d'uso con altri moduli

Questa classe utilizzerà le seguenti classi:

- View::Pages::Execution;
- Model::MongoRelations::Loader::LoaderClass;

Attributi

Al momento non sono stati previsti degli attributi.

Metodi

- Execution(string id): costruttore che dovrà inizializzare gli eventuali attibuti e la pagina View::Pages::Execution in base all'id della presentazione. Execution(string id) controllerà la presenza del token di sessione, usando il metodo goIndex() in caso negativo, e dialogherà con Model::MongoRelations::Loader::LoaderClass per caricare la presentazione dal database;
- +next(): metodo che viene invocato premendo il tasto "freccia destra" e che invoca a sua volta il metodo impress().next() implementato all'interno del framework Impress.js e che visualizzerà il frame successivo della presentazione;
- +prev(): metodo che viene invocato premendo il tasto "freccia sinistra" e che invoca a sua volta il metodo impress().prev() implementato all'interno del framework Impress.js e che visualizzerà il frame precedente della presentazione;
- +bookmark(): metodo che viene invocato premendo il tasto "barra spaziatrice" e che invoca a sua volta il metodo impress().bookmark() implementato all'interno del framework Impress.js e che visualizzerà il frame con bookmark successivo;
- +goHome(): metodo che reindirizza alla pagina View::Pages::Home;
- +goIndex(): metodo che reindirizza alla pagina View::Pages::Index;
- +goEdit(): metodo che reindirizza alla pagina View::Pages::Edit.

6.5 Controller::EditController

Funzione

Questa classe si occuperà di mostrare all'utente la possibilità di apportare modifiche ad una presentazione.

Relazioni d'uso con altri moduli

Questa classe utilizzerà le seguenti classi:

- View::Pages::Edit;

- Model::SlideShow::SlideShowActions::Command:
  - ConcreteTextInsertCommand;
  - ConcreteFrameInsertCommand;
  - ConcreteImageInsertCommand;
  - ConcreteSVGInsertCommand;
  - ConcreteAudioInsertCommand;
  - ConcreteVideoInsertCommand;
  - ConcreteBackgroundInsertCommand;
  - ConcreteTextRemoveCommand;
  - ConcreteFrameRemoveCommand;
  - ConcreteImageRemoveCommand;
  - ConcreteSVGRemoveCommand;
  - ConcreteAudioRemoveCommand;
  - ConcreteVideoRemoveCommand;
  - ConcreteBackgroundRemoveCommand;
  - ConcreteEditSizeCommand;
  - ConcreteEditPositionCommand;
  - ConcreteEditRotationCommand;
  - ConcreteEditColorCommand;
  - ConcreteEditBackgroundCommand;
  - ConcreteEditFontCommand;
  - ConcreteEditContentCommand; [[[[[[[[[[[[[[[[ESISTE????]]]]]]]]]]]]]]]]
  - Invoker;
- Model::ApacheManager::FileManager;
- Model::ApacheRelations::ResourceGetter;
- Model::MongoRelations::Loader::Autenticazione;
- Model::MongoRelations::Loader::Loaderclass;

Attributi

Al momento non sono stati previsti degli attributi.

Metodi

- +Edit(string id): costruttore che dovrà inizializzare gli eventuali attibuti e la pagina View::Pages::Edit in base all'id della presentazione. Edit(string id) controllerà la presenza del token di sessione, usando il metodo goIndex() in caso negativo, e dialogherà con Model::MongoRelations::Loader::LoaderClass per caricare la presentazione dal database;

- +insertFrame(string shape, int posX, int posY): metodo che dialoga con Model::SlideShow::SlideShow passandogli le informazioni per l'inserimento di un nuovo frame, la sua forma e le sue coordinate. Se l'inserimento avviene correttamente, View::Pages::Edit verrà aggiornata col nuovo frame inserito, altrimenti verrà segnalato un errore;
- +insertMedia(string path, int posX, int posY): metodo che dialoga con ConcreteImageInsertCommand, ConcreteVideoInsertCommand e ConcreteAudioInsertCommand (facenti parte di Model::SlideShow::SlideShowActions::Command) comunicando il percorso del file media in locale in modo da caricarlo sul server e le sue coordinate di posizione sul piano della presentazione. Se l'inserimento avviene correttamente, View::Pages::Edit verrà aggiornata col nuovo file media inserito, altrimenti verrà segnalato un errore;
- +moveElement(string id, int posX, int posY): metodo che dialoga con Model::SlideShow::SlideShowAc passando l'id dell'elemento spostato e le sue nuove coordinate di posizione. Se la modifica avviene correttamente, View::Pages::Edit verrà aggiornata con l'elemento nella nuova posizione, altrimenti verrà segnalato un errore;
- +insertText(string text, int posX, int posY): metodo che dialoga con Model::SlideShow::SlideShowAct comunicando la stringa del testo e le coordinate di posizione sul piano della presentazione. Se l'inserimento avviene correttamente, View::Pages::Edit verrà aggiornata col nuovo elemento testo inserito, altrimenti verrà segnalato un errore;
- +textEdit(string id, string text): metodo che dialoga con Model::SlideShow::SlideShowActions::Comm passando l'id dell'elemento di testo e la nuova stringa associata. Se la modifica avviene correttamente, View::Pages::Edit verrà aggiornata con l'elemento di testo aggiornato con la nuova stringa, altrimenti verrà segnalato un errore;
- +deleteElement(string id): metodo che dialoga con ConcreteTextRemoveCommand, ConcreteFrameRemoveCommand, ConcreteImageRemoveCommand, ConcreteSVGRemoveCommand, ConcreteAudioRemoveCommand e ConcreteVideoRemoveCommand (facenti parte di Model::SlideShow::SlideShowActions::Command) comunicando l'id dell'elemento da rimuovere dal piano della presentazione. Se la rimozione avviene correttamente, View::Pages::Edit verrà aggiornata col nuovo file media inserito, altrimenti verrà segnalato un errore;
- +insertChoice(string id, string choiceId): [[[[[[[[[[[[[[[[[[VEDEREEEEEEE]]]]]]]]]]]]]]]]];
- +bookmark(string id): [[[[[[[[[[[[[[[[[[VEDEREEEEEE]]]]]]]]]]]]]]]]];
- +changeSize(string id, int zoom): metodo che dialoga con Model::SlideShow::SlideShowActions::Com passando l'id dell'elemento da ingrandire e il nuovo valore di zoom da applicare. Se la modifica avviene correttamente, View::Pages::Edit verrà aggiornata con l'elemento zoomato al valore indicato, altrimenti verrà segnalato un errore;
- +changeRotation(string id, int rotation): metodo che dialoga con Model::SlideShow::SlideShowAction passando l'id dell'elemento da ruotare e il nuovo valore di rotazione da applicare. Se la modifica avviene correttamente, View::Pages::Edit verrà aggiornata con l'elemento ruotato al valore indicato, altrimenti verrà segnalato un errore;
- +changePath(array id): [[[[[[[[[[[[[[VEDEREEEEEE]]]]]]]]]]]]]];

- +frameBackground(string id, string path): metodo che dialoga con Model::SlideShow::SlideShowActi passando l'id del frame e il percorso dell'immagine da impostare come nuovo sfondo del frame. Se la modifica avviene correttamente, View::Pages::Edit verrà aggiornata con lo sfondo nuovo, altrimenti verrà segnalato un errore;
- +slideshowBackground(string path): metodo che dialoga con Model::SlideShow::SlideShowActions::C passando il percorso dell'immagine da impostare come nuovo sfondo della presentazione. Se la modifica avviene correttamente, View::Pages::Edit verrà aggiornata con lo sfondo nuovo, altrimenti verrà segnalato un errore;
- +insertSvg(string shape, string color, int posX, int posY): metodo che dialoga con Model::SlideShow::SlideShowActions::Command::ConcreteSVGInsertCommand comunicando il tipo di SVG da inserire, le sue coordinate di posizione sul piano della presentazione e il suo colore. Se l'inserimento avviene correttamente, View::Pages::Edit verrà aggiornata col nuovo SVG, altrimenti verrà segnalato un errore;
- +esegui(string id): metodo che richiama la funzione goExecute(id);
- +goExecute(string id): reindirizza alla pagina View::Pages::Execute passando l'id della presentazione da eseguire;
- +goHome(): metodo che reindirizza alla pagina View::Pages::Home;
- +goIndex(): metodo che reindirizza alla pagina View::Pages::Index.

7 Package Premi::View

Tipo, obiettivo e funzione del componente: contiene le classi che istanzieranno gli oggetti per l'interfaccia grafica del software.

Relazioni d'uso di altre componenti: utilizza le classi contenute nel package Premi::Controller per le comunicazioni con il Model.

Attività svolte e dati trattati: rappresenta l'intera GUI del nostro sistema.

7.1 Premi::View::Pages

Tipo, obiettivo e funzione del componente: contiene le pagine in Html, rappresentano l'interfaccia grafica vera e propria.

Relazioni d'uso di altre componenti: utilizza le classi contenute nel package Premi::Controller.

Attività svolte e dati trattati: rappresenta le pagine fisiche del software.

7.2 Premi::View::Pages::Index

Funzione

Questa pagina si occuperà di mostrare all'utente la possibilità di effettuare il login oppure di registrarsi al sistema.

Relazioni d'uso con altri moduli

Questa pagina utilizzerà le seguenti classi:

- Premi::Controller::IndexController

Attributi

Al momento non sono stati previsti degli attributi.

Input utente

- bottoneLogin(): attiva il metodo Premi::Controller::IndexController::login(), se non sono compilati entrambi i campi restituisce errore, se va tutto bene reindirizza alla pagina Premi::View::Pages::Home;
- bottoneSubscription(): attiva il metodo Premi::Controller::IndexController::subscription(), se non sono compilati entrambi i campi restituisce errore, se va tutto bene reindirizza alla pagina Premi::View::Pages::Home.

7.3 Premi::View::Pages::Home

Funzione

Questa pagina si occuperà di mostrare all'utente le presentazioni presenti sul proprio database e per ognuno di esse darà la possibilità di eliminarle, eseguirle, scaricarle in locale,

rinominarle, modificarle, crearne una nuova o effettuare il logout.

Relazioni d'uso con altri moduli
Questa pagina utilizzerà le seguenti classi:

- Premi::Controller::HomeController

Attributi
Al momento non sono stati previsti degli attributi.

Input utente

- bottoneDeleteSlideShow(): bottone che fa scomparire il riferimento alla presentazione di cui il bottone fa riferimento, attiva il metodo di Premi::Controller::HomeController che si occuperà dell'eliminazione della presentazione dal database;
- bottoneDownload(): bottone che attiva il Premi::Controller::HomeController che si occuperà dello scaricamento della presentazione nel Manifest e manda a schermo un segnale di avvenuto scaricamento della presentazione;
- bottoneRenameSlideShow(): bottone che attiva il Premi::Controller::HomeController che si occuperà della rinominazione della presentazione nel database e manda a schermo l'effettva modifica del nome della presentazione;
- bottoneExecute(): bottone che reindirizza alla pagina Premi::View::Pages::Execution e attiva il controller Premi::Controller::HomeController che si prende l'id della presentazione alla quale il bottone fa riferimento per la creazione della prossima pagina;
- bottoneEdit(): bottone che reindirizza alla pagina Premi::View::Pages::Edit e attiva il controller Premi::Controller::HomeController che si prende l'id della presentazione alla quale il bottone fa riferimento per la creazione della prossima pagina;
- bottoneLogout(): bottone che attiva il Premi::Controller::HomeController che si occuperà della distruzione del token di sessione e reindirizza alla pagina Premi::View::Pages::Index.

7.4 Premi::View::Pages::Profile

Funzione
Questa pagina si occuperà di mostrare all'utente la possibilità di cambiare i propri dati personali, caricare, eliminare o rinominare i propri file media.

Relazioni d'uso con altri moduli
Questa pagina utilizzerà le seguenti classi:

- Premi::Controller::ProfileController

Attributi
Al momento non sono stati previsti degli attributi.

Input utente

- bottoneChangePassword(): bottone che, se i campi sono compilati erroneamente restituisce errore, altrimenti attiva il metodo in Premi::Controller::ProfileController che si occupa di cambiare la password dell'account e restituisce un messaggio a schermo di avvenuto cambiamento;
- bottoneUploadMedia(): bottone che attiva il metodo in Premi::Controller::ProfileController che caricherà il file richiesto nel server, se l'operazione va a buon fine la schermata si aggiorna con la nuova miniatura del file media;
- bottoneDeleteMedia(): bottone che fa scomparire il riferimento al file media di cui il bottone fa riferimento, attiva il metodo di Premi::Controller::ProfileController che si occuperà dell'eliminazione del file media dal server;
- bottoneRenameMedia(): bottone che attiva il Premi::Controller::ProfileController che si occuperà della rinominazione del file media presente nel server e manda a schermo l'effettva modifica del nome del file media.

7.5 Premi::View::Pages::Execution

Funzione

Questa pagina si occuperà di gestire l'esecuzione di una presentazione utilizzando il framework Impress.js associato alla pagina.

Relazioni d'uso con altri moduli

Questa pagina utilizzerà le seguenti classi:

- Premi::Controller::ExecutionController

Attributi

Al momento non sono stati previsti degli attributi.

Input utente

- +next(): metodo che viene invocato premendo il tasto "freccia destra" e che invoca a sua volta il metodo impress().next() implementato all'interno del framework Impress.js e che visualizzerà il frame successivo della presentazione;
- +prev(): metodo che viene invocato premendo il tasto "freccia sinistra" e che invoca a sua volta il metodo impress().prev() implementato all'interno del framework Impress.js e che visualizzerà il frame precedente della presentazione;
- +bookmark(): metodo che viene invocato premendo il tasto "barra spaziatrice" e che invoca a sua volta il metodo impress().bookmark() implementato all'interno del framework Impress.js e che visualizzerà il frame con bookmark successivo.

7.6 Premi::View::Pages::Edit

Funzione

Questa pagina si occuperà di mostrare all'utente la possibilità di apportare modifiche ad

una presentazione.

Relazioni d'uso con altri moduli

Questa pagina utilizzerà le seguenti classi:

- Premi::Controller::EditController

Attributi

Al momento non sono stati previsti degli attributi.

Input utente

- bottoneInsertFrame(): bottone che fa comparire nel piano della presentazione il nuovo frame in base alla forma selezionata, e attiva il metodo in Premi::Controller::EditController che si occuperà di aggiornare le informazioni della presentazione;
- bottoneInsertMedia(): bottone che fa comparire nel piano della presentazione il nuovo file media e attiva il metodo in Premi::Controller::EditController che aggiornerà le informazioni della presentazione e caricherà il file richiesto nel server;
- dropMove(): evento che si attiva al rilascio dello spostamento di un elemento, attiva il metodo in Premi::Controller::EditController che si occuperà dell'aggiornamento delle informazioni della presentazione;
- bottoneInsertText(): bottone che fa comparire nel piano della presentazione il nuovo elemento testuale e attiva il metodo in Premi::Controller::EditController che aggiornerà le informazioni della presentazione;
- bottoneTextEdit(): bottone che modifica il testo selezionato e attiva il metodo in Premi::Controller::EditController che si occupa dell'aggiornamento dell'elemento testuale modificato;
- bottoneDeleteElement(): bottone che fa scomparire l'elemento dal piano della presentazione e attiva il metodo di Premi::Controller::EditController che si occuperà dell'aggiornamento della presentazione;
- bottoneInsertChoice(): bottone che fa inserire all'utente il testo della scelta e fa scegliere il frame al quale farà riferimento, attiva il metodo di Premi::Controller::EditController che si occuperà dell'aggiornamento della presentazione;
- bottoneBookmark(): bottone che assegna o rimuove il bookmark al frame, attiva il metodo di Premi::Controller::EditController che si occuperà dell'aggiornamento della presentazione;
- dropSize(): evento che si attiva al rilascio del ridimensionamento di un elemento, attiva il metodo in Premi::Controller::EditController che si occuperà dell'aggiornamento delle informazioni della presentazione;
- dropRotation(): evento che si attiva al rilascio della rotazione di un elemento, attiva il metodo in Premi::Controller::EditController che si occuperà dell'aggiornamento delle informazioni della presentazione;

- bottonePath(): bottone che mostra all'utente l'ordine dei frame e da la possibilità di modificarlo, se viene modificato attiva il metodo in Premi::Controller::EditController che si occuperà dell'aggiornamento delle informazioni della presentazione;
- bottoneFrameBackground(): bottone assegna l'immagine caricata come sfondo del frame e che attiva il metodo in Premi::Controller::EditController per caricare il file nel server e lo imposta come sfondo del frame;
- bottoneBackground(): bottone che assegna l'immagine caricata come sfondo della presentazione e attiva il metodo in Premi::Controller::EditController per caricare il file nel server e lo imposta come sfondo della presentazione;
- bottoneInsertSvg(): bottone che fa comparire nel piano della presentazione il nuovo elemento svg selezionato e attiva il metodo in Premi::Controller::EditController che si occuperà di aggiornare le informazioni della presentazione;
- bottoneExecute(): bottone che reindirizza alla pagina Premi::View::Pages::Execute e attiva il metodo in Premi::Controller::EditController che salverà l'id della presentazione da eseguire nella pagina successiva;