













- Premi::Model::ConcreteRemoveCommand -> invoca i metodi per eliminare un elemento di tipo testo dalla presentazione.

Interfacce con e relazioni d'uso e da altre componenti: Viene invocato per eliminare elementi testuali in una presentazione.

Classi ereditate:

- Premi::Model::Eliminazione::Remover.

5.1.2.3 Premi::Model::Eliminazione:: ConcreteFrameRemover

Tipo, obiettivo e funzione del componente: Classe che implementa un algoritmo di eliminazione di un elemento di tipo frame all'interno di una presentazione. È uno dei componenti concreti del Design Pattern Template.

Relazioni d'uso di altre componenti:

- Premi::Model::ConcreteRemoveCommand -> invoca i metodi per eliminare un elemento di tipo frame dalla presentazione.

Interfacce con e relazioni d'uso e da altre componenti: Viene invocato per eliminare elementi di tipo frame da una presentazione.

Classi ereditate:

- Premi::Model::Eliminazione::Remover.

5.1.2.4 Premi::Model::Eliminazione:: ConcreteSVGtRemover

Tipo, obiettivo e funzione del componente: Classe che implementa un algoritmo di eliminazione di un elemento di tipo SVG all'interno di una presentazione. È uno dei componenti concreti del Design Pattern Template.

Relazioni d'uso di altre componenti:

- Premi::Model::ConcreteRemoveCommand -> invoca i metodi per eliminare un elemento di tipo SVG dalla presentazione.

Interfacce con e relazioni d'uso e da altre componenti: Viene invocato per eliminare elementi SVG da una presentazione.

Classi ereditate:

- Premi::Model::Eliminazione::Remover.

5.1.2.5 Premi::Model::Eliminazione:: ConcreteImageRemover

Tipo, obiettivo e funzione del componente: Classe che implementa un algoritmo di eliminazione di un elemento immagine all'interno di una presentazione. È uno dei componenti concreti del Design Pattern Template.

Relazioni d'uso di altre componenti:

- Premi::Model::ConcreteRemoveCommand -> invoca i metodi per eliminare un elemento di tipo immagine dalla presentazione.



5.1.3.3 Premi::Model::Modifica::EditorSize

Tipo, obiettivo e funzione del componente: Classe concreta del Design Pattern Strategy per la modifica dei campi inerenti alla dimensione di un elemento della presentazione.

Relazioni d'uso di altre componenti:

- Premi::Model::Command::ConcreteEditCommand -> Invoca il costruttore di EditorSize;
- Premi::Model::Presentazione::SlideShow<- scorre gli elementi dei membri contenitori all'interno di SlideShow per trovare l'elemento da modificare;
- Premi::Model::Presentazione::SlideShowElement <- invoca le funzioni della classe SlideShowElement per modificare opportunamente i campi relativi alla dimensione dell'elemento.

Interfacce con e relazioni d'uso e da altre componenti: Premi::Model::Command::ConcreteEditCommand invoca il costruttore e la funzione di esecuzione dell'operazione di modifica, EditorSize invocherà quindi i metodi di modifica delle dimensioni forniti all'interno della sottoclasse di Premi::Model::Presentazione::SlideShowElement di cui fa parte l'oggetto da modificare.

Classi ereditate:

- Premi::Model::Modifica::Editor.

5.1.3.4 Premi::Model::Modifica::EditorRotate

Tipo, obiettivo e funzione del componente: Classe concreta del Design Pattern Strategy per la modifica dei campi inerenti all'inclinazione di un elemento della presentazione.

Relazioni d'uso di altre componenti:

- Premi::Model::Command::ConcreteEditCommand -> Invoca il costruttore di EditorRotate;
- Premi::Model::Presentazione::SlideShow<- scorre gli elementi dei membri contenitori all'interno di SlideShow per trovare l'elemento da modificare;
- Premi::Model::Presentazione::SlideShowElement <- invoca le funzioni della classe SlideShowElement per modificare opportunamente i campi relativi all'inclinazione dell'elemento.

Interfacce con e relazioni d'uso e da altre componenti: Premi::Model::Command::ConcreteEditCommand invoca il costruttore e la funzione di esecuzione dell'operazione di modifica, EditorPosition invocherà quindi i metodi di modifica dell'inclinazione forniti all'interno della sottoclasse di Premi::Model::Presentazione::SlideShowElement di cui fa parte l'oggetto da modificare.

Classi ereditate:

- Premi::Model::Modifica::Editor.



- **Premi::Controller::Presentazione::EditController** -> crea un oggetto della classe **Premi::Model::Command** passandolo all'Invoker che lo esegue (execute) e lo inserisce nello stack "undo", richiama il metodo che svuota lo stack "redo". Può inoltre invocare il metodo "unexecute" dell'Invoker che provvede a richiamare il metodo undo del comando sulla cima dello stack "undo" e a spostarlo quindi nello stack "redo". Alternativamente invoca il metodo "redo" dell'Invoker che provvede a eseguire il comando sulla cima dello stack "redo" e a spostarlo quindi nello stack "undo".

Interfacce con e relazioni d'uso e da altre componenti: Viene invocato per effettuare le operazioni di modifica alla presentazione, a sua volta invoca una classe derivata da **Premi::Model::Command** per eseguire materialmente il comando. Quando un comando viene eseguito, Invoker lo salva in un array \$undo[], insieme ai parametri necessari a riportare la presentazione allo stato precedente.

5.1.4.2 **Premi::Model::Command::AbstractCommand**

Tipo, obiettivo e funzione del componente: È interfaccia astratta del Design Pattern Command, è classe base per i comandi di modifica, inserimento ed eliminazione.

Relazioni d'uso di altre componenti:

- **Premi::Model::Invoker** -> esegue materialmente il comando, richiamandone i metodi di esecuzione; inoltre provvede ad annullare l'ultima operazione

Interfacce con e relazioni d'uso e da altre componenti: Viene utilizzata per applicare un generico parametro di trasformazione ad un oggetto della presentazione, questo parametro verrà poi specificato dalle classi concrete.

Sottoclassi:

- **Premi::Model::Command::ConcreteInsertCommand;**
- **Premi::Model::Command::ConcreteRemoveCommand;**
- **Premi::Model::Command::ConcreteEditCommand.**

5.1.4.3 **Premi::Model::Command::ConcreteInsertCommand**

Tipo, obiettivo e funzione del componente: È classe concreta del Design Pattern Command, rappresenta un comando per inserire un nuovo elemento nell'oggetto presentazione.

Relazioni d'uso di altre componenti:

- **Premi::Controller::Presentazione::EditController** -> invoca il costruttore della classe e passa l'oggetto così creato all'Invoker; **Premi::Model::Invoker** -> esegue il comando o ne invoca il metodo di annullamento;
- **Premi::Model::Inserimento::Inserter** <- invoca la classe concreta del template per l'inserimento di un elemento.

Interfacce con e relazioni d'uso e da altre componenti: Viene utilizzata per gestire i Signal riguardanti l'inserimento di un nuovo elemento ed invocare i corretti metodi del Model;
Classi ereditate:

- **Premi::Model::Command::AbstractCommand.**





5.2 Controller

Tipo, obiettivo e funzione del componente: fanno parte di questo livello i package che gestiscono i segnali e le chiamate effettuati dalla view verso la struttura dati.

Relazioni d'uso di altre componenti: il componente è costituito dai package Presentazione e Utente, comunica con il Model per rendere possibile la gestione del profilo e la gestione delle presentazioni da parte dell'utente.

Package contenuti:

- Premi::Controller::Presentazione
- Premi::Controller::Utente

5.2.1 Premi::Controller::Presentazione

Tipo, obiettivo e funzione del componente: fanno parte di questo package tutte le classi di controller con cui interagiscono le pagine dedicate alla gestione delle presentazioni.

Relazioni d'uso di altre componenti: il package comunica con la view ricevendo chiamate da Premi::View::Pages::MobileEdit, Premi::View::Pages::DesktopEdit, Premi::View::Pages::Execution e Premi::View::Pages::Home. Comunica, invece, con il model inviando segnali e chiamate ai package Premi::Model::Inserimento, Premi::Model::Eliminazione, Premi::Model::Modifica, Premi::Model::Command, Premi::Model::Builder e alle classi Premi::Model::Invoker, Premi::Model::MongoHan

5.2.1.1 Premi::Controller::Presentazione::EditController

Tipo, obiettivo e funzione del componente: Lo scopo di questa classe è di gestire i segnali delle pagine Premi::View::Pages::DesktopEdit e Premi::View::Pages::MobileEdit verso il model.

Relazioni d'uso di altre componenti:

- Premi.View.Pages.DesktopEdit e Premi.View.Pages.MobileEdit -> costruiscono EditController, ne invocano i metodi passando i parametri degli oggetti modificati;
- Premi::Model::Command <- EditController costruisce un comando e lo dà in pasto a Premi::Model::Invoker;
- Premi::Model::Invoker <- EditController costruisce l'oggetto di classe Invoker. Invoca il metodo execute() di Invoker, passando come parametro un oggetto di classe Command oppure invoca il metodo unexecute() di Invoker.

Interfacce con e relazioni d'uso e da altre componenti: La pagina DesktopEdit o la pagina MobileEdit invia a EditController un segnale comunicando l'avvenuta modifica o la rimozione di un elemento della presentazione, oppure l'inserimento di un nuovo elemento. EditController istanzia un oggetto di classe Premi::Model::Command e lo dà in pasto a Premi::Model::Invoker. Eventualmente EditController può semplicemente annullare il comando appena eseguito invocando il metodo unexecute di Invoker.



presentazioni.

Relazioni d'uso di altre componenti: la classe Home, utilizza i metodi messi a disposizione dalle seguenti classi presenti nel package Controller:

EDITORICA EDITOR [|||||] per caricare la presentazione da modificare;

INSERIMENTO [|||||] per l'inserimento di nuovi elementi;

POSTAMENTO [|||||] per lo spostamento di nuovi elementi;

ELIMINAZIONE [|||||] per l'eliminazione elementi;

MODIFICA ELEMENTI [|||||] per le modifiche effettuate agli elementi ;

CAMBIAMENTO PERCORSO [|||||] per cambiare il percorso della presentazione.

Attività svolte e dati trattati: La classe definisce la struttura della pagina web che consente agli utenti di modificare una presentazione (inserendo, spostando, modificando o eliminando elementi), cambiare il percorso, assegnare bookmark ai frame e inserire elementi scelta.

5.3.6 Premi.View.Pages.MobileEdit

Tipo, obiettivo e funzione del componente: la classe MobileEdit definisce la struttura, e la conseguente visualizzazione, della pagina web che mostra ad un utente mobile l'editor di modifica mobile di una presentazione.

Relazioni d'uso di altre componenti: la classe MobileEdit utilizza i metodi messi a disposizione dalle seguenti classi presenti nel package Controller:

EDITOR MOBILE [|||||] per caricare la presentazione da modificare;

INSERIMENTO TESTO [|||||] per l'inserimento di un elemento testuale;

MODIFICA TESTO [||||] per la modifica di un elemento testuale;

INSERIMENTO BOOKMARK [||||] per l'inserimento di un nuovo bookmark;

ELIMINAZIONE BOOKMARK [||||] per rimuovere un bookmark.

Attività svolte e dati trattati: La classe definisce la struttura della pagina web che consente agli utenti di modificare una presentazione (modificando un elemento testo) e assegnare bookmark ai frame..