



Obelix Group
obelixswe@gmail.com

Definizione di Prodotto

Versione	<i>v1_0_0</i>
Data creazione	2017-05-01
Redattori	
Verificatori	
Approvazione	
Stato	Approvato
Uso	esterno
Distribuzione	Prof. Tullio Vardanega Prof. Riccardo Cardin Red Babel Gruppo Obelix

Sommario

Questo documento descrive l'architettura generale e in dettaglio del prodotto Monolith che verrà sviluppato dal gruppo Obelix.



Diario delle revisioni

Versione	Modifica	Autore e Ruolo	Data
2.0.0	Approvazione Documento	Tomas Mali Responsabile	2017-04-29
0.4.1	Verifica sezione Diagrammi Attività	Riccardo Saggese Verificatore	2017-05-14
0.4.0	Verifica sezione Diagrammi di Sequenza	Federica Schifano Verificatore	2017-05-15
0.3.0	Verifica sezione Diagrammi Attività	Riccardo Saggese Verificatore	2017-05-14
0.2.0	Verifica sezione Standard di Progetto	Federica Schifano Verificatore	2017-05-12
0.1.0	Verifica prime tre sezioni	Riccardo Saggese Verificatore	2017-05-12
0.0.10	Stesura sottosezione Voto sondaggio	Riccardo Saggese Progettista	2017-05-11
0.0.9	Stesura sottosezione Cancellazione bolla voto sondaggio	Riccardo Saggese Progettista	2017-05-10
0.0.8	Stesura sottosezione Creazione bolla	Riccardo Saggese Progettista	2017-05-09
0.0.7	Inizio stesura sezione Diagrammi di sequenza	Riccardo Saggese Progettista	2017-05-08
0.0.6	Stesura sottosezione Configurazione ListBubble	Silvio Meneguzzo Progettista	2017-05-07
0.0.5	Stesura sottosezione Configurazione sondaggio	Federica Schifano Progettista	2017-05-06
0.0.4	Inizio stesura sezione Diagrammi Attività	Federica Schifano Progettista	2017-05-05
0.0.3	Stesura sezione Standard di Progetto	Riccardo Saggese Progettista	2017-05-05
0.0.4	Inizio stesura sezione Descrizione Architettuale	Silvio Meneguzzo Progettista	2017-05-04
0.0.3	Stesura sezione Tecnologie Utilizzate	Emanuele Crespan Progettista	2017-05-03
0.0.2	Stesura sezione Introduzione	Federica Schifano Progettista	2017-05-01



Versione	Modifica	Autore e Ruolo	Data
0.0.1	Creato template	Nicolò Rigato Progettista	2017-05-01



Indice

1	Introduzione	10
1.1	Scopo del documento	10
1.2	Scopo del prodotto	10
1.3	Glossario	10
1.4	Riferimenti	10
1.4.1	Normativi	10
1.4.2	Informativi	10
2	Tecnologie Utilizzate	11
2.1	Javascript 6th edition (ECMA SCRIPT 6)	11
2.2	Meteor	11
2.3	Mongo DB	12
2.4	HTML5	12
2.5	SCSS	13
2.6	React	13
2.7	Node.js	14
2.8	Rocket.chat	15
2.9	Bootstrap	15
2.10	polyglot.js	16
2.11	Money.js	16
2.12	weather.js	17
2.13	classNames	17
3	Descrizione Architettura	17
3.1	Metodo e formalismo di specifica	17
3.2	Architettura generale - Componenti del sistema	18
3.2.1	Monolith	18
3.2.2	Monolith::Database	19
3.2.3	Monolith::Database::InformationStorage	20
3.2.4	Monolith::Database::informationStorage::Checks	21
3.2.5	Monolith::Database::InformationStorage::DatabaseSettings	22
3.2.6	Monolith::UI	23
3.2.7	Monolith::UI::Bubbles	24
3.2.8	Monolith::UI::SideAreas	26
3.2.9	Monolith::UI::SideAreas::SideArea1_pkg	27
3.2.10	Monolith::UI::SideAreas::SideArea2_pkg	28
3.2.11	Monolith::UI::UI-Layouts	29
3.2.12	Monolith::UI::UI-SingleComponents	30
3.3	Architettura generale - Bolle Demo	32
3.3.1	CurrencyBubble	32
3.3.2	DiceBubble	34
3.3.3	ListBubble	36
3.3.4	ListBubble::CheckListCreation	37
3.3.5	ListBubble::CheckListReading	39
3.3.6	ListBubble::Configuration	40
3.3.7	ListBubble::DataManagement	41
3.3.8	ListBubble::Receiver	42
3.3.9	ListBubble::Sender	43



3.3.10	MeteoBubble	44
3.3.11	SurveyBubble	46
3.3.12	TranslationBubble	48
3.4	Architettura di dettaglio - Classi del sistema Monolith	50
3.4.1	check	50
3.4.2	checkCreator	51
3.4.3	concreteCheck	52
3.4.4	checkDiscriminator	53
3.4.5	concreteCheckCreator	54
3.4.6	Bubble	55
3.4.7	BubbleConfig	56
3.4.8	bubbleCreator	57
3.4.9	ConcreteBubble	58
3.4.10	bubbleDiscriminator	59
3.4.11	BubbleCreationButton	60
3.4.12	ConcreteBubbleConfig	61
3.4.13	concreteBubbleCreator	62
3.4.14	ConcreteBubbleCreationButton	63
3.4.15	SideArea1	64
3.4.16	SentBubbleHistory	65
3.4.17	BubbleCreationMenu	66
3.4.18	SideArea2	67
3.4.19	ReceivedBubbleHistory	68
3.4.20	VerticalLayout	69
3.4.21	ContainedElement	70
3.4.22	HorizontalLayout	71
3.4.23	ConditionalRendering	72
3.4.24	Image	73
3.4.25	ComboBox	74
3.4.26	LineEdit	75
3.4.27	LabelEdit	76
3.4.28	PushButton	77
3.4.29	CheckButton	78
3.4.30	ImageButton	79
3.4.31	CheckBoxList	80
3.4.32	LabelComboBox	81
3.4.33	TextAreaButton	82
3.4.34	LabelPushButton	83
3.4.35	LineEditComboBox	84
3.4.36	RadioButtonGroup	85
3.4.37	TextAreaComboBox	86
3.4.38	LineEditPushButton	87
3.4.39	LabelEditPushButton	88
3.5	Architettura di dettaglio - Classi delle bolle demo	88
3.5.1	CurrencyConversion	88
3.5.2	CurrencyBubbleSender	90
3.5.3	CurrencyBubbleCreator	91
3.5.4	CurrencyBubbleReceiver	92
3.5.5	CurrencyBubbleConfigMenu	93
3.5.6	DiceRoller	94



3.5.7	DiceBubbleSender	95
3.5.8	DiceBubbleCreator	96
3.5.9	DiceBubbleReceiver	97
3.5.10	DiceBubbleConfigMenu	98
3.5.11	CheckListCreator	99
3.5.12	CheckListComponent	100
3.5.13	CheckListItemsDefinition	101
3.5.14	CheckList	102
3.5.15	ListOfCheckLists	103
3.5.16	ListCreationButton	104
3.5.17	ListBubbleMenuConfig	105
3.5.18	ListBubbleCreator	106
3.5.19	ListBubbleReceiver	107
3.5.20	ListBubbleSender	108
3.5.21	MeteoItem	109
3.5.22	MeteoDelivery	110
3.5.23	MeteoBubbleSender	111
3.5.24	MeteoBubbleCreator	112
3.5.25	MeteoBubbleReceiver	113
3.5.26	MeteoBubbleConfigMenu	114
3.5.27	ResultsViewer	115
3.5.28	SurveyManager	116
3.5.29	SurveyBubbleSender	117
3.5.30	SurveyBubbleCreator	118
3.5.31	SurveyBubbleReceiver	119
3.5.32	SurveyBubbleConfigMenu	120
3.5.33	MessageTranslation	121
3.5.34	TranslationBubbleSender	122
3.5.35	TranslationBubbleCreator	123
3.5.36	TranslationBubbleReceiver	124
3.5.37	TranslationBubbleConfigMenu	125
4	Standard di Progetto	125
4.1	Standard di documentazione del codice	125
4.2	Standard di denominazione di entità e relazioni	126
4.3	Strumenti di lavoro	126
5	Diagrammi di Attività	126
5.1	Configurazione sondaggio	126
5.2	Configurazione ListBubble	127
6	Diagramma di Sequenza	128
6.1	Creazione Bolla	128
6.2	Cancellazione Bolla	129
6.3	Voto Bolla-Sondaggio	130



7	Tracciamento	130
7.1	Tracciamento componenti-requisiti	130
7.2	Tracciamento requisiti-componenti	131
7.3	Tracciamento classi-requisiti	131
7.4	Tracciamento requisiti-classi	131
A	Descrizione Design Pattern	131
A.1	Design Pattern Utilizzati	132
A.1.1	Factory Method	132



Elenco delle figure

1	Diagramma per Monolith.	18
2	Diagramma per Monolith::Database::InformationStorage.	20
3	Diagramma per Monolith::Database::informationStorage::Checks.	21
4	Diagramma per Monolith::UI::Bubbles.	24
5	Diagramma per Monolith::UI::UI-Layouts.	29
6	Diagramma per Monolith::UI::UI-SingleComponents.	30
7	Diagramma per CurrencyBubble.	32
8	Diagramma per DiceBubble.	34
9	Diagramma per ListBubble.	36
10	Diagramma per ListBubble::CheckListCreation.	37
11	Diagramma per ListBubble::CheckListReading.	39
12	Diagramma per ListBubble::Configuration.	40
13	Diagramma per ListBubble::DataManagement.	41
14	Diagramma per ListBubble::Receiver.	42
15	Diagramma per ListBubble::Sender.	43
16	Diagramma per MeteoBubble.	44
17	Diagramma per SurveyBubble.	46
18	Diagramma per TranslationBubble.	48
19	Diagramma per check in Checks	50
20	Diagramma per checkCreator in Checks	51
21	Diagramma per concreteCheck in Checks	52
22	Diagramma per checkDiscriminator in Checks	53
23	Diagramma per concreteCheckCreator in Checks	54
24	Diagramma per Bubble in Bubbles	55
25	Diagramma per BubbleConfig in Bubbles	56
26	Diagramma per bubbleCreator in Bubbles	57
27	Diagramma per ConcreteBubble in Bubbles	58
28	Diagramma per bubbleDiscriminator in Bubbles	59
29	Diagramma per BubbleCreationButton in Bubbles	60
30	Diagramma per ConcreteBubbleConfig in Bubbles	61
31	Diagramma per concreteBubbleCreator in Bubbles	62
32	Diagramma per ConcreteBubbleCreationButton in Bubbles	63
33	Diagramma per SideArea1 in SideArea1_pkg	64
34	Diagramma per SentBubbleHistory in SideArea1_pkg	65
35	Diagramma per BubbleCreationMenu in SideArea1_pkg	66
36	Diagramma per SideArea2 in SideArea2_pkg	67
37	Diagramma per ReceivedBubbleHistory in SideArea2_pkg	68
38	Diagramma per VerticalLayout in UI-Layouts	69
39	Diagramma per ContainedElement in UI-Layouts	70
40	Diagramma per HorizontalLayout in UI-Layouts	71
41	Diagramma per ConditionalRendering in UI-Layouts	72
42	Diagramma per Image in UI-SingleComponents	73
43	Diagramma per ComboBox in UI-SingleComponents	74
44	Diagramma per LineEdit in UI-SingleComponents	75
45	Diagramma per LabelEdit in UI-SingleComponents	76
46	Diagramma per PushButton in UI-SingleComponents	77
47	Diagramma per CheckButton in UI-SingleComponents	78
48	Diagramma per ImageButton in UI-SingleComponents	79



49	Diagramma per CheckBoxList in UI-SingleComponents	80
50	Diagramma per LabelComboBox in UI-SingleComponents	81
51	Diagramma per TextAreaButton in UI-SingleComponents	82
52	Diagramma per LabelPushButton in UI-SingleComponents	83
53	Diagramma per LineEditComboBox in UI-SingleComponents	84
54	Diagramma per RadioButtonGroup in UI-SingleComponents	85
55	Diagramma per TextAreaComboBox in UI-SingleComponents	86
56	Diagramma per LineEditPushButton in UI-SingleComponents	87
57	Diagramma per LabelEditPushButton in UI-SingleComponents	88
58	Diagramma per CurrencyConversion in	89
59	Diagramma per CurrencyBubbleSender in	90
60	Diagramma per CurrencyBubbleCreator in	91
61	Diagramma per CurrencyBubbleReceiver in	92
62	Diagramma per CurrencyBubbleConfigMenu in	93
63	Diagramma per DiceRoller in	94
64	Diagramma per DiceBubbleSender in	95
65	Diagramma per DiceBubbleCreator in	96
66	Diagramma per DiceBubbleReceiver in	97
67	Diagramma per DiceBubbleConfigMenu in	98
68	Diagramma per CheckListCreator in CheckListCreation	99
69	Diagramma per CheckListComponent in CheckListCreation	100
70	Diagramma per CheckListItemsDefinition in CheckListCreation	101
71	Diagramma per CheckList in CheckListReading	102
72	Diagramma per ListOfCheckLists in CheckListReading	103
73	Diagramma per ListCreationButton in Configuration	104
74	Diagramma per ListBubbleMenuConfig in Configuration	105
75	Diagramma per ListBubbleCreator in DataManagement	106
76	Diagramma per ListBubbleReceiver in Receiver	107
77	Diagramma per ListBubbleSender in Sender	108
78	Diagramma per MeteoItem in	109
79	Diagramma per MeteoDelivery in	110
80	Diagramma per MeteoBubbleSender in	111
81	Diagramma per MeteoBubbleCreator in	112
82	Diagramma per MeteoBubbleReceiver in	113
83	Diagramma per MeteoBubbleConfigMenu in	114
84	Diagramma per ResultsViewer in	115
85	Diagramma per SurveyManager in	116
86	Diagramma per SurveyBubbleSender in	117
87	Diagramma per SurveyBubbleCreator in	118
88	Diagramma per SurveyBubbleReceiver in	119
89	Diagramma per SurveyBubbleConfigMenu in	120
90	Diagramma per MessageTranslation in	121
91	Diagramma per TranslationBubbleSender in	122
92	Diagramma per TranslationBubbleCreator in	123
93	Diagramma per TranslationBubbleReceiver in	124
94	Diagramma per TranslationBubbleConfigMenu in	125
95	Diagramma di attività per la bolla sondaggio	127
96	Diagramma di attività per la bolla ListBubble	128
97	Diagramma di sequenza per la creazione di una bolla	128
98	Diagramma di sequenza per la cancellazione di una bolla	129



99	Diagramma di sequenza per il voto in una bolla-sondaggio	130
100	Diagramma del Factory method	132

Elenco delle tabelle

3	Tracciamento componenti - requisiti	130
5	Tracciamento requisiti - componenti	131
7	Tracciamento classi - requisiti	131
9	Tracciamento requisiti - classi	131



1 Introduzione

1.1 Scopo del documento

Questo documento ha come scopo quello di definire la progettazione ad alto e a basso livello per il prodotto Monolith. Verrà presentata l'architettura generale secondo la quale saranno organizzate le varie componenti software e i Design Pattern utilizzati nella creazione dell'SDK, delle bolle predefinite e della demo. Verrà inoltre dettagliato il tracciamento tra le componenti software individuate ed i requisiti.

1.2 Scopo del prodotto

Lo scopo del prodotto è quello di permettere la creazione di bolle interattive, che dovranno funzionare nell'ambiente Rocket.chat. Queste bolle permetteranno di aumentare l'interattività tra gli utenti della chat e aggiungeranno nuove funzionalità accessibili direttamente dalla conversazione senza il bisogno di ricorrere all'apertura di applicazioni diverse. Il sistema offrirà agli sviluppatori un set di $API_{|G|}$ per creare e rilasciare nuove bolle e agli utenti finali la possibilità di usufruire di un insieme di bolle predefinite.

1.3 Glossario

Al fine di evitare ogni ambiguità di linguaggio e massimizzare la comprensione dei documenti, i termini che necessitano di essere chiariti saranno scritti in corsivo e marcati con una $|G|$ in pedice alla prima occorrenza e saranno riportati nel Glossario.

1.4 Riferimenti

1.4.1 Normativi

- **Norme di Progetto:**
NormediProgetto_v1.1.0
- **Capitolato d'appalto C5:**
<http://www.math.unipd.it/~tullio/IS-1/2016/Progetto/C5.pdf>
- **Analisi dei Requisiti:**
Analisi dei Requisiti_v1.0.0

1.4.2 Informativi

- **Slide del corso di Ingegneria del Software:**
<http://www.math.unipd.it/~tullio/IS-1/2016/>
- **Documentazione React**
- **Documentazione Meteor**
- **Documentazione ECMAScript 6**
- **Libro Design Patterns** Design Patterns, Elementi per il riuso di software a oggetti. Gamma, Helm, Johnson, Vlissides.



2 Tecnologie Utilizzate

In questa sezione verranno descritte le tecnologie su cui si basa lo sviluppo del progetto. Per ognuna di esse, verranno indicati l'ambito di utilizzo della tecnologia, i vantaggi e gli svantaggi che ne derivano. Alcune delle tecnologie che saranno usate sono richieste come requisito dal capitolato scelto.

2.1 Javascript 6th edition (ECMA SCRIPT 6)

JavaScript è un linguaggio di scripting orientato agli oggetti e agli eventi. È comunemente utilizzato nella programmazione Web lato client per la creazione, in siti web e applicazioni web, di effetti dinamici interattivi tramite l'uso di funzioni di script invocate da eventi innescati in vari modi dall'utente sulla pagina web in uso.

Come richiesto dal capitolato, per la realizzazione di Monolith, deve essere utilizzato Javascript 6th edition (ECMA SCRIPT 6).

Licenza Non esiste una sola implementazione perché ECMAScript (o ES) è un linguaggio di programmazione standardizzato e mantenuto da Ecma International nell'ECMA-262 ed ISO/IEC 16262.

Vantaggi

- Gestione degli eventi asincroni tramite le promises
- Possibilità di dichiarare classi
- Supporto per le costanti (*const*)
- Possibilità di isolare la definizione di variabili ad un blocco (*let*)
- Possibilità di isolare lo scope di una funzione usando blocchi delimitati da parentesi graffe() come ambienti isolati (vs closure)
- Uso di sintassi più espressiva per scrivere le funzioni anonime (*Arrow Functions*)

Svantaggi

- Il supporto di ES6 da parte dei browser è ancora incompleto
- L'assenza di tipizzazione potrebbe ostacolare la valutazione della correttezza del codice

2.2 Meteor

Meteor è un framework web JavaScript libero e open source per lo sviluppo di applicazioni web e mobile. È una piattaforma basata su Node.js. Meteor utilizza, dunque, JavaScript sia lato client che lato server.



Licenza MIT La licenza MIT è una delle licenze più permissive nel panorama open source. In modo più esplicito dichiara i diritti dati all'utente finale, incluso il diritto di utilizzare, copiare, modificare, incorporare, pubblicare, distribuire, sotto-licenziare, e/o vendere il software.

Vantaggi

- Integrazione con diverse tecnologie utilizzate nello sviluppo web:
 - React
 - MongoDB
- Isomorfismo: il codice javascript scritto funziona in modo trasparente sul client (browser), sul server (Node.js) o in entrambi i mondi
- Ecosistema e modularità: la comunità di Meteor è molto attiva e molte funzionalità client o server potrebbero già essere pacchettizzate dal package manager ufficiale.

Svantaggi

- Inizialmente sconosciuto ai membri del gruppo.

2.3 Mongo DB

MongoDB è un database NoSQL orientato ai documenti, basato sul formato BSON per la memorizzazione e la rappresentazione dei dati. È distribuito come software libero open source.

Licenza

 GNU AGPL v3.0

È una licenza pubblicata da Free Software Foundation. È simile alla capostipite GNU GPL, una licenza fortemente copyleft per software libero.

Vantaggi

- È più flessibile di un database SQL e facilita la rappresentazione su un modello ad oggetti
- Supporta ricerche per campi, intervalli e regular expression. Le query possono restituire campi specifici del documento e anche includere funzioni definite dall'utente in JavaScript.
- Qualunque campo in MongoDB può essere indicizzato

Svantaggi

- Inizialmente sconosciuto ai membri del gruppo.

2.4 HTML5

HTML5 è un linguaggio di markup per la strutturazione delle pagine web.



Licenza Non esiste una sola implementazione perché HTML5 è un linguaggio di markup standardizzato e mantenuto da W3C.

Vantaggi

- Codice più pulito e sintassi semplificata rispetto alle versioni precedenti
- Interattività senza l'ausilio di plugin esterni valida per diversi formati multimediali
- Semantica intuitiva grazie ai nuovi TAG di formattazione
- Introduzione della geolocalizzazione, dovuta ad una forte espansione di sistemi operativi mobili
- Sistema più efficiente alternativo ai normali cookie chiamato Web Storage

Svantaggi

- Non tutti i browser supportano HTML5

2.5 SCSS

SCSS è una sintassi per i fogli di stile introdotta da Sass 3 (Syntactically Awesome StyleSheets). È un'estensione del CSS .

Licenza MIT

La licenza MIT è una delle licenze più permissive nel panorama open source. In modo più esplicito dichiara i diritti dati all'utente finale, incluso il diritto di utilizzare, copiare, modificare, incorporare, pubblicare, distribuire, sotto-licenziare, e/o vendere il software.

Vantaggi

- Possibilità di utilizzare variabili
- Possibilità di creare funzioni
- Possibilità di organizzare il foglio di stile in più file
- Compatibilità completa con la sintassi del CSS

Svantaggi

- Sintassi più complessa.

2.6 React

React è una libreria Javascript open source che permette di costruire interfacce utente.

**Licenza** BSD-3-Clause

Le licenze BSD sono una famiglia di licenze permissive, senza copyleft, per software. Le tre clausole della licenza BSD-3-Clause sono:

- Libertà di eseguire il programma per qualsiasi scopo
- Libertà di studiare il programma e modificarlo
- Libertà di ridistribuire copie del programma in modo da aiutare il prossimo

Vantaggi

- Semplificazione della realizzazione di interfacce UI dinamiche che possono reagire ai cambiamenti di dati in maniera autonoma attraverso opportuni componenti
- Possibilità di utilizzare le viste per creare codice più facile da comprendere e su cui è più semplice effettuare il debugging.

Svantaggi

- Implementa solo la parte puramente visuale dell'applicazione. Esistono alternative che possono gestire molti più aspetti.
- Curva di apprendimento ripida
- È una libreria relativamente nuova

2.7 Node.js

Node.js è una piattaforma event-driven per il motore JavaScript V8. Essa permette di realizzare applicazioni web utilizzando il linguaggio JavaScript, che tipicamente è usato client-side, per la scrittura anche della parte server-side delle applicazioni web.

Licenza MIT

La licenza MIT è una delle licenze più permissive nel panorama open source. In modo più esplicito dichiara i diritti dati all'utente finale, incluso il diritto di utilizzare, copiare, modificare, incorporare, pubblicare, distribuire, sotto-licenziare, e/o vendere il software.

Vantaggi

- Facile apprendimento
- Possibilità di realizzare applicazioni server-side senza dover utilizzare linguaggi di programmazione “tradizionali”

Svantaggi

- Non supporta database relazionali



2.8 Rochet.chat

Rocket.chat è una Web chat server sviluppata in Javascript utilizzando il *Framework*_[G] Meteor.

Licenza MIT

La licenza MIT è una delle licenze più permissive nel panorama open source. In modo più esplicito dichiara i diritti dati all'utente finale, incluso il diritto di utilizzare, copiare, modificare, incorporare, pubblicare, distribuire, sotto-licenziare, e/o vendere il software.

Vantaggi

- Codice open source
- Possibilità di creare chat di gruppo
- Possibilità di inviare audio, video e file
- Possibilità di effettuare video chiamate
- Community molto attiva

Svantaggi

- Parzialmente documentata

2.9 Bootstrap

Bootstrap è una raccolta di strumenti liberi per la creazione di siti e applicazioni per il Web. Essa contiene modelli di progettazione basati su HTML e CSS, sia per la tipografia, che per le varie componenti dell'interfaccia, come moduli, pulsanti e navigazione, così come alcune estensioni opzionali di JavaScript.

Licenza MIT

La licenza MIT è una delle licenze più permissive nel panorama open source. In modo più esplicito dichiara i diritti dati all'utente finale, incluso il diritto di utilizzare, copiare, modificare, incorporare, pubblicare, distribuire, sotto-licenziare, e/o vendere il software.

Vantaggi

- Piattaforma ben standardizzata
- Non richiede l'appoggio né di un linguaggio di programmazione server side, né di un database
- Ottima documentazione
- Responsive Design
- É supportato dai browser moderni

**Svantaggi**

- I plugin di jQuery sono limitati
- Le modifiche dovute al continuo sviluppo non sono sempre facili da integrare

2.10 polyglot.js

Polyglot.js è una libreria per la traduzione scritta in JavaScript, eseguita sia per il browser che per gli ambienti CommonJS(Node).

Licenza MIT

La licenza MIT è una delle licenze più permissive nel panorama open source. In modo più esplicito dichiara i diritti dati all'utente finale, incluso il diritto di utilizzare, copiare, modificare, incorporare, pubblicare, distribuire, sotto-licenziare, e/o vendere il software.

Vantaggi

- Non è richiesta una iscrizione per l'utilizzo
- È una libreria non a pagamento
- Il Polyglot ha zero dipendenze
- Copre una traduzione di 30 lingue diverse

2.11 Money.js

Money.js è una libreria semplice con l'unico obiettivo di convertire un valore di denaro da qualsiasi valuta in qualsiasi altra valuta. Money.js utilizza una fusione algoritmica per calcolare un insieme di tassi costantemente preciso per 165 valute mondiali.

Licenza MIT

La licenza MIT è una delle licenze più permissive nel panorama open source. In modo più esplicito dichiara i diritti dati all'utente finale, incluso il diritto di utilizzare, copiare, modificare, incorporare, pubblicare, distribuire, sotto-licenziare, e/o vendere il software.

Vantaggi

- Non è richiesta una iscrizione per l'utilizzo
- È una libreria non a pagamento
- È una libreria semplice da integrare nel codice JavaScript



2.12 weather.js

Weather.js è una libreria che recupera i dati da openweathermap.org e fa la ricerca di tutti i tipi di informazioni relative alle condizioni meteo.

Licenza MIT

La licenza MIT è una delle licenze più permissive nel panorama open source. In modo più esplicito dichiara i diritti dati all'utente finale, incluso il diritto di utilizzare, copiare, modificare, incorporare, pubblicare, distribuire, sotto-licenziare, e/o vendere il software.

Vantaggi

- Non è richiesta una iscrizione per l'utilizzo
- É una libreria non a pagamento
- É una libreria semplice da integrare nel codice JavaScript

Svantaggi

- Ha bisogno di 11 dipendenze

2.13 classNames

classNames è una semplice utility raccomandata per l'uso con React per l'unione condizionata di className

Licenza MIT

La licenza MIT è una delle licenze più permissive nel panorama open source. In modo più esplicito dichiara i diritti dati all'utente finale, incluso il diritto di utilizzare, copiare, modificare, incorporare, pubblicare, distribuire, sotto-licenziare, e/o vendere il software.

Vantaggi

- Semplifica la gestione dei className dinamici
- Non possiede ulteriori dipendenze

3 Descrizione Architettura

3.1 Metodo e formalismo di specifica

Nell'esposizione dell'architettura dell'applicazione si procederà con un approccio top-down, descrivendo l'architettura iniziando dal generale ed andando al particolare. Si procederà quindi alla descrizione dei package, per poi descrivere nel dettaglio le singole classi, specificando per ognuna il tipo, l'obiettivo, la funzione e le relazioni in ingresso ed in uscita. Successivamente si illustreranno



degli esempi di uso dei Design Pattern nell'architettura del sistema, rimandando la spiegazione generale alla sezione dedicata. L'architettura dell' SDK e della demo sono state progettate separatamente. Per i diagrammi delle componenti di classe e di attività, si utilizza il formalismo UML 2.0. Le classi e componenti presenti in librerie o framework esterni vengono contraddistinte da colori diversi. I framework esterni verranno rappresentati con un colore azzurro, mentre le classi e componenti proprie invece, saranno rappresentate con un colore giallo. Nella demo le componenti dell'SDK sono in verde. L'intera applicazione è progettata utilizzando il framework *Meteor*_{|G|} che permette di utilizzare il linguaggio JavaScript sia per il lato client che per quello server (tramite NodeJS). I diagrammi delle classi che permettono di mostrare l'architettura generale del sistema vengono affiancati anche dai diagrammi di sequenza e attività, che permettono di definire le interazioni tra le componenti, senza preoccuparsi della loro classificazione.

3.2 Architettura generale - Componenti del sistema

3.2.1 Monolith

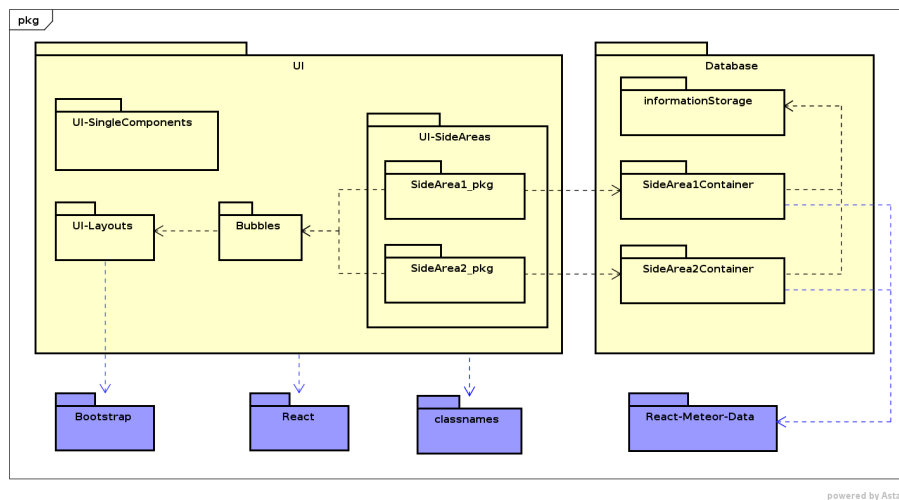


Figura 1: Diagramma per Monolith.

Descrizione:

Componente che rappresenta l'intera SDK di Monolith



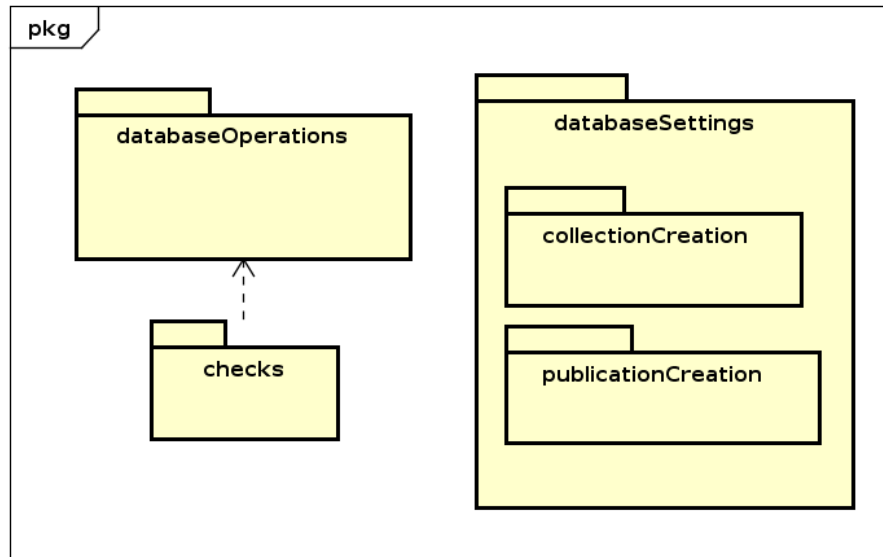
3.2.2 Monolith::Database

Descrizione:

Componente contenente i pacchetti che vengono utilizzati per interagire con il database



3.2.3 Monolith::Database::InformationStorage



powered by Astah

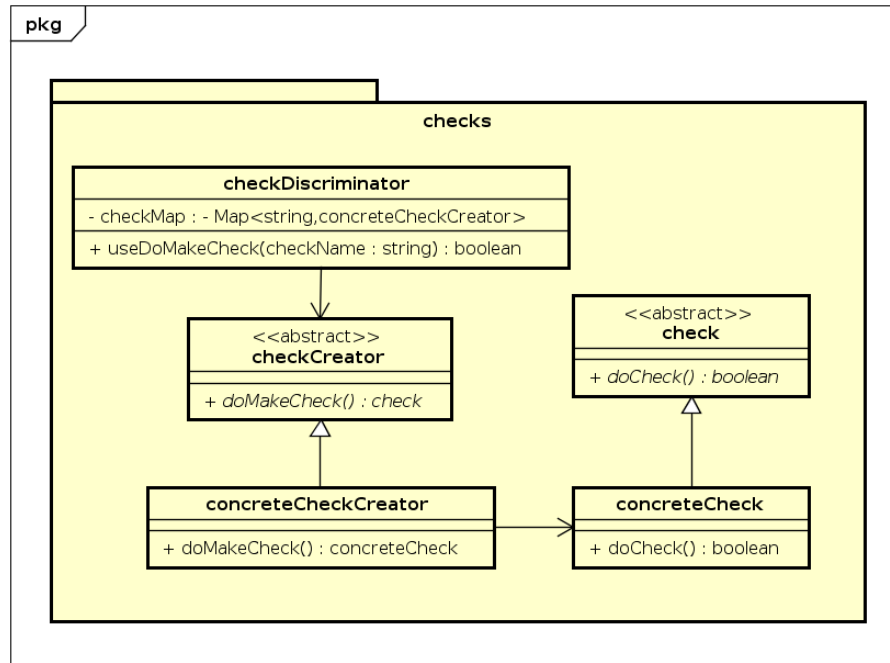
Figura 2: Diagramma per Monolith::Database::InformationStorage.

Descrizione:

Componente per la configurazione dell'utilizzo del database



3.2.4 Monolith::Database::informationStorage::Checks



powered by Astah

Figura 3: Diagramma per Monolith::Database::informationStorage::Checks.

Descrizione:

Componente per la creazione dei controlli da effettuare sul client prima di effettuare l'inserimento dei dati nel database

Classi contenute:

- check
- checkCreator
- checkDiscriminator
- concreteCheck
- concreteCheckCreator



3.2.5 Monolith::Database::InformationStorage::DatabaseSettings

Descrizione:

Modulo per la configurazione delle collection



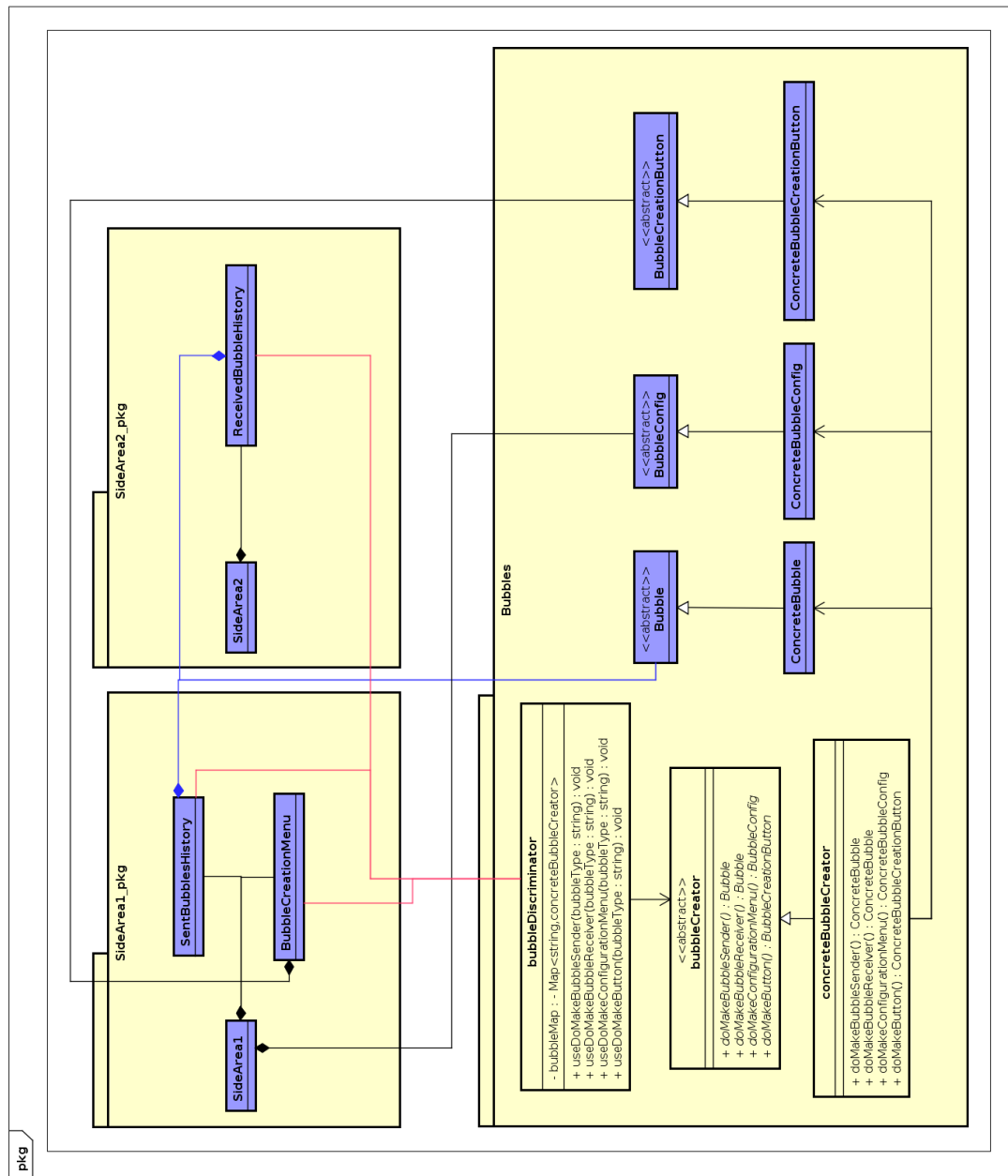
3.2.6 Monolith::UI

Descrizione:

Componente contenente tutti i pacchetti che servono per comporre e gestire la parte visuale dell'applicazione delle bolle **Dipendenze**

- React
- classNames

3.2.7 Monolith::UI::Bubbles



powered by Astah

Figura 4: Diagramma per Monolith::UI::Bubbles.

Descrizione:

Componente per la creazione delle bolle da visualizzare

Classi contenute:

- Bubble
- BubbleConfig



- BubbleCreationButton
- bubbleCreator
- bubbleDiscriminator
- ConcreteBubble
- ConcreteBubbleConfig
- ConcreteBubbleCreationButton
- concreteBubbleCreator



3.2.8 Monolith::UI::SideAreas

Descrizione:

Contiene i package per la visualizzazione delle bolle nelle side-bar



3.2.9 Monolith::UI::SideAreas::SideArea1_pkg

Descrizione:

Componente per la visualizzazione delle bolle inviate e del menù di creazione delle bolle nella prima side-bar

Classi contenute:

- BubbleCreationMenu
- SentBubbleHistory
- SideArea1



3.2.10 Monolith::UI::SideAreas::SideArea2_pkg

Descrizione:

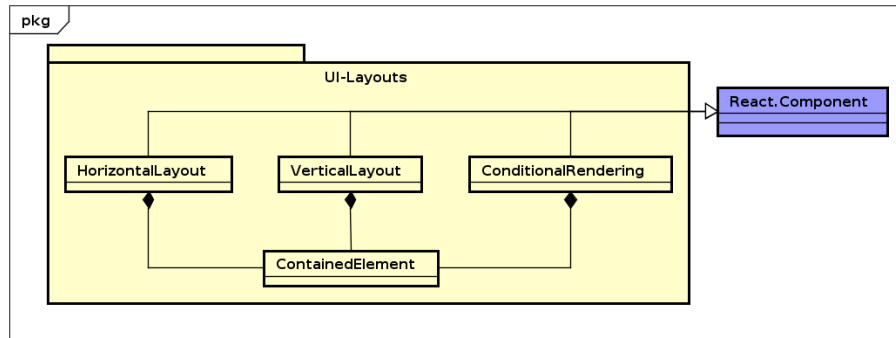
Componente per la visualizzazione delle bolle ricevute nella seconda side-bar

Classi contenute:

- ReceivedBubbleHistory
- SideArea2



3.2.11 Monolith::UI::UI-Layouts



powered by Astah

Figura 5: Diagramma per Monolith::UI::UI-Layouts.

Descrizione:

Componente che contiene le classi React per la gestione dei layout **Dipendenze**
Bootstrap

Classi contenute:

- ConditionalRendering
- ContainedElement
- HorizontalLayout
- VerticalLayout



3.2.12 Monolith::UI::UI-SingleComponents

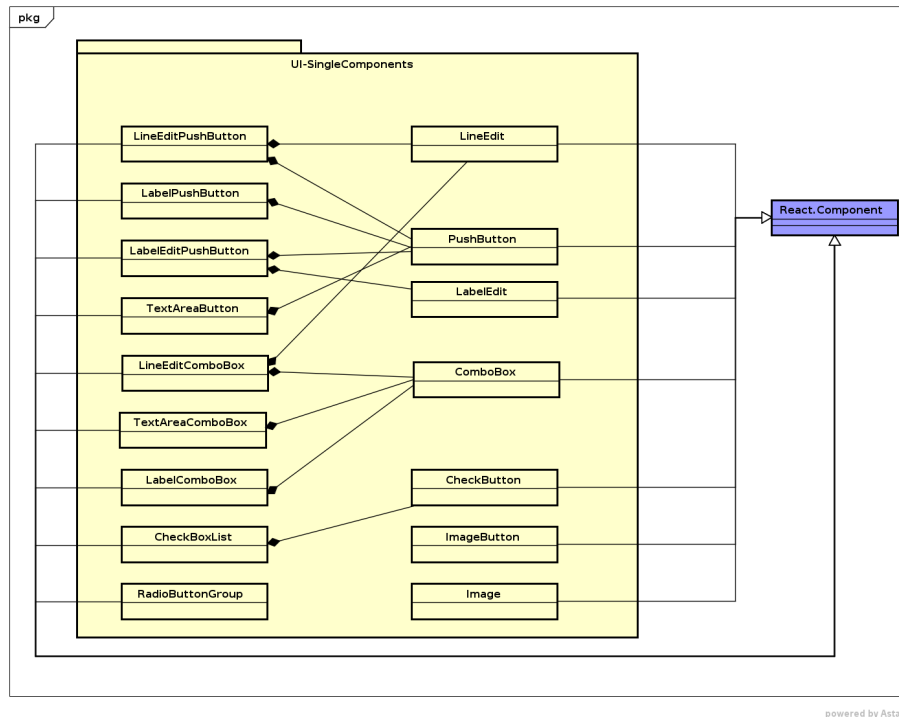


Figura 6: Diagramma per Monolith::UI::UI-SingleComponents.

Descrizione:

Componente che contiene tutti i componenti React per la composizione della GUI

Classi contenute:

- CheckBoxList
- CheckButton
- ComboBox
- Image
- ImageButton
- LabelComboBox
- LabelEdit
- LabelEditPushButton
- LabelPushButton
- LineEdit



- LineEditComboBox
- LineEditPushButton
- PushButton
- RadioButtonGroup
- TextAreaButton
- TextAreaComboBox

3.3 Architettura generale - Bolle Demo

3.3.1 CurrencyBubble

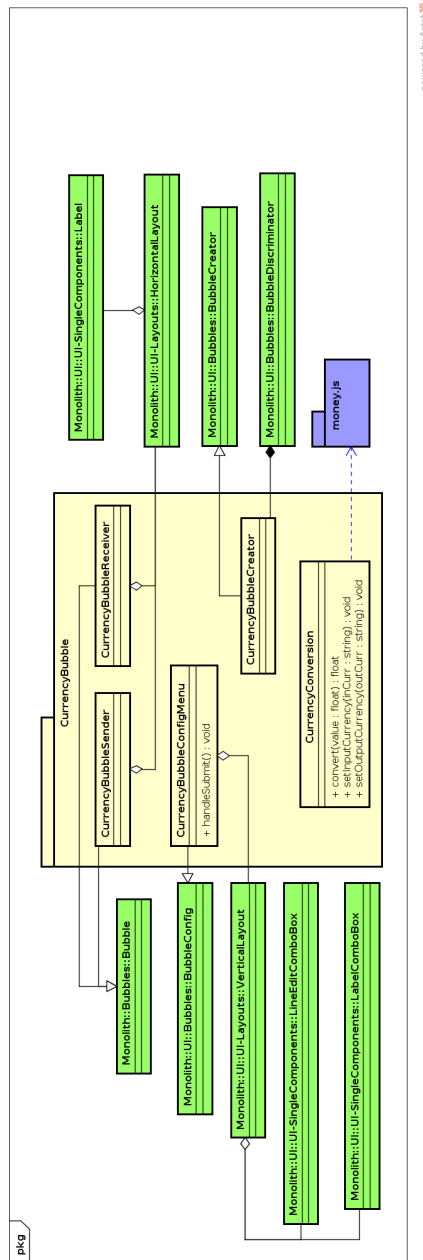


Figura 7: Diagramma per CurrencyBubble.

Descrizione:

Componente contenente le classi necessarie per la creazione della bolla convertitore valute.

Classi contenute:

- CurrencyBubbleConfigMenu



- CurrencyBubbleCreator
- CurrencyBubbleReceiver
- CurrencyBubbleSender
- CurrencyConversion



3.3.2 DiceBubble

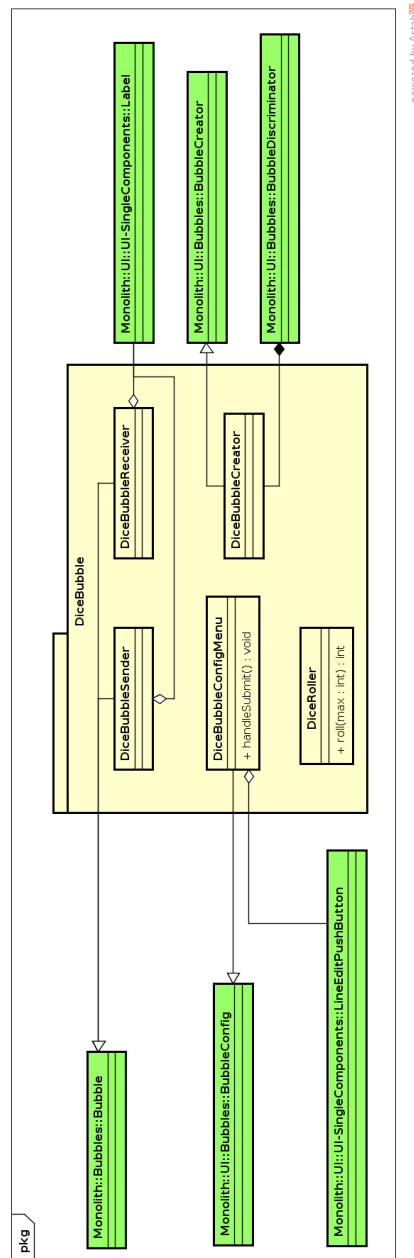


Figura 8: Diagramma per DiceBubble.

Descrizione:

Componente contenente le classi necessarie per la creazione della bolla estrazione numero casuale.

Classi contenute:

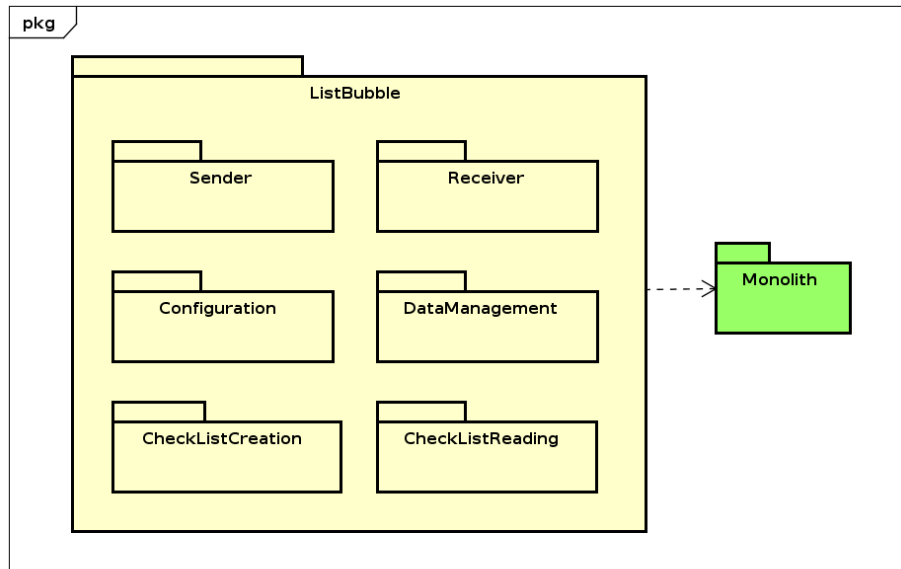
- DiceBubbleConfigMenu
- DiceBubbleCreator



- DiceBubbleReceiver
- DiceBubbleSender
- DiceRoller



3.3.3 ListBubble



powered by Astah

Figura 9: Diagramma per ListBubble.

Descrizione:

Componente contenente i pacchetti necessari per la creazione della bolla lista.

3.3.4 ListBubble::CheckListCreation

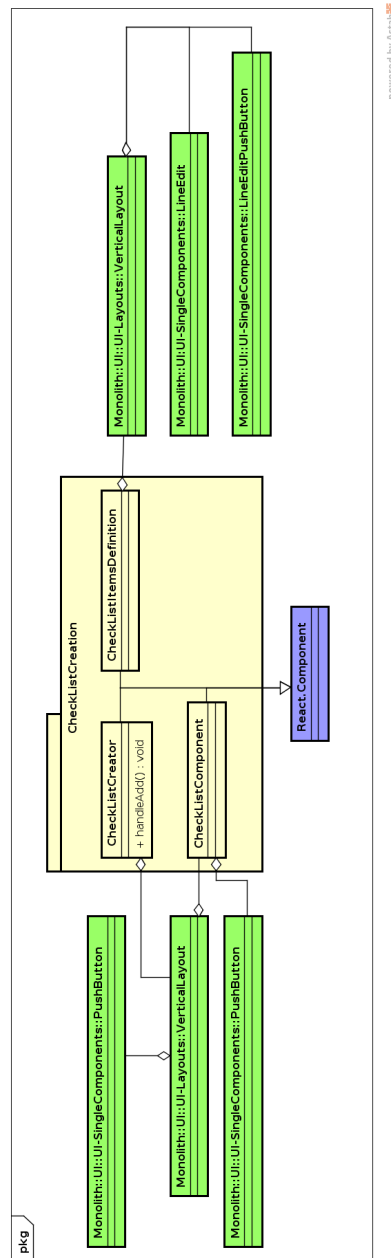


Figura 10: Diagramma per ListBubble::CheckListCreation.

Descrizione:

Componente che si occupa della creazione delle check list.

Classi contenute:

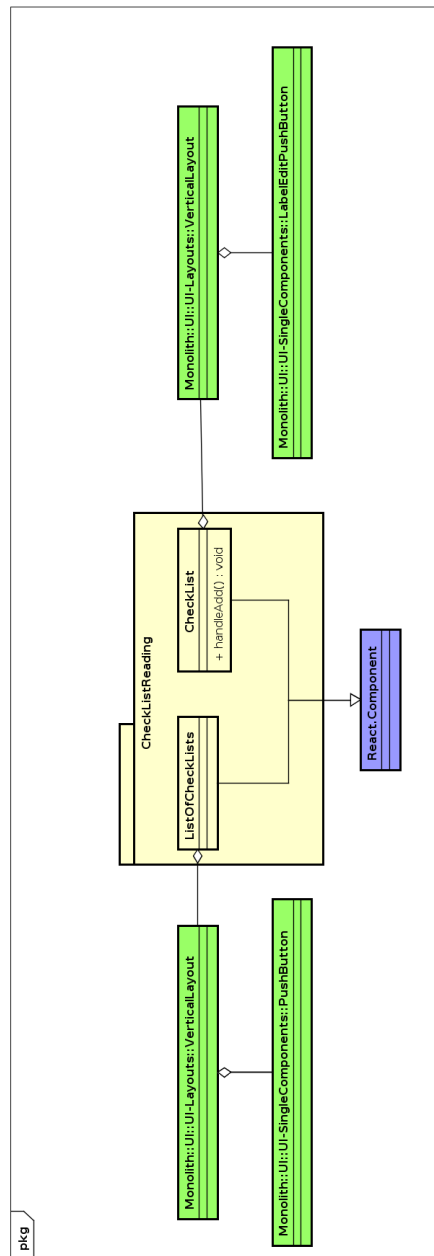
- CheckListComponent
- CheckListCreator



- CheckListItemsDefinition



3.3.5 ListBubble::CheckListReading



powered by Astah

Figura 11: Diagramma per ListBubble::CheckListReading.

Descrizione:

Componente che si occupa della lettura e dell'utilizzo delle check list.

Classi contenute:

- CheckList
- ListOfCheckLists

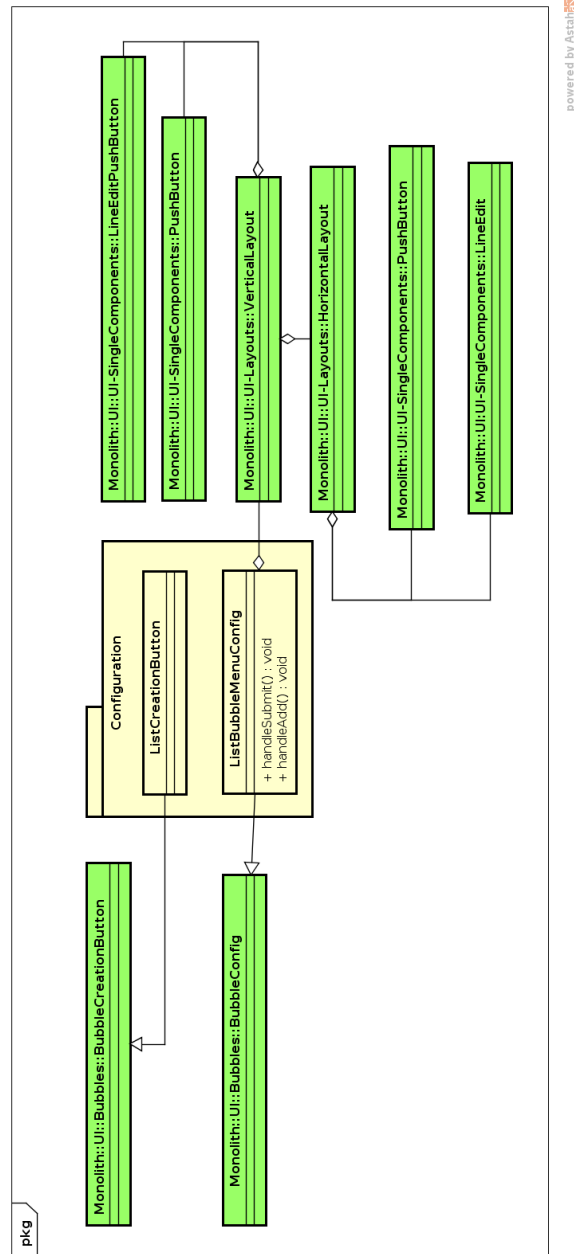
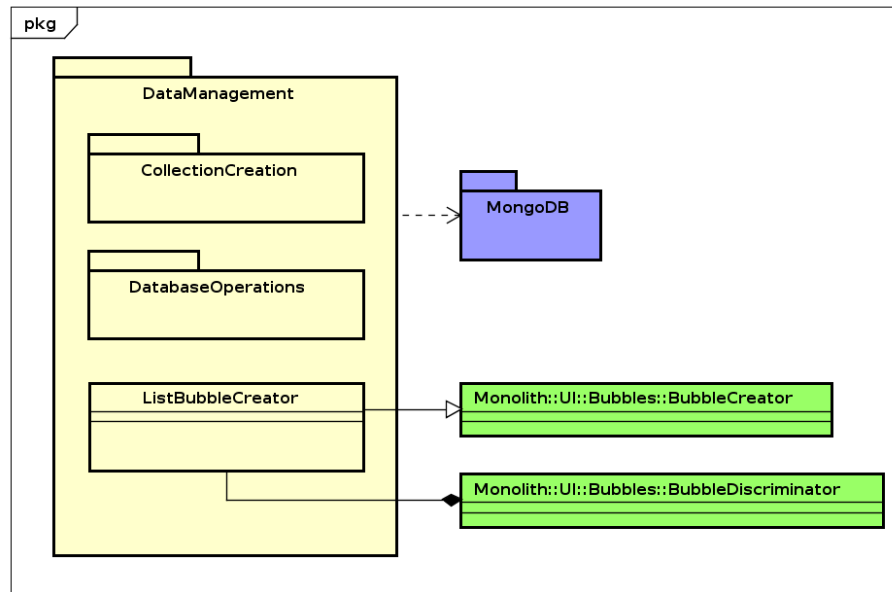


Figura 12: Diagramma per ListBubble::Configuration.

- ListBubbleMenuConfig
- ListCreationButton



3.3.7 ListBubble::DataManagement



powered by Astah

Figura 13: Diagramma per ListBubble::DataManagement.

Descrizione:

Componente che si occupa di tutte le operazioni di gestione dei dati che non sono gestite da Monolith. Usa il database MongoDB.

Classi contenute:

- ListBubbleCreator



3.3.8 ListBubble::Receiver

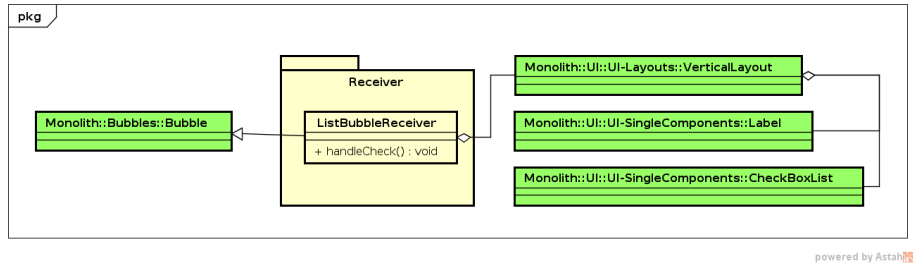


Figura 14: Diagramma per ListBubble::Receiver.

Descrizione:

Componente che gestisce la visualizzazione della bolla da parte del ricevente.

Classi contenute:

- ListBubbleReceiver



3.3.9 ListBubble::Sender

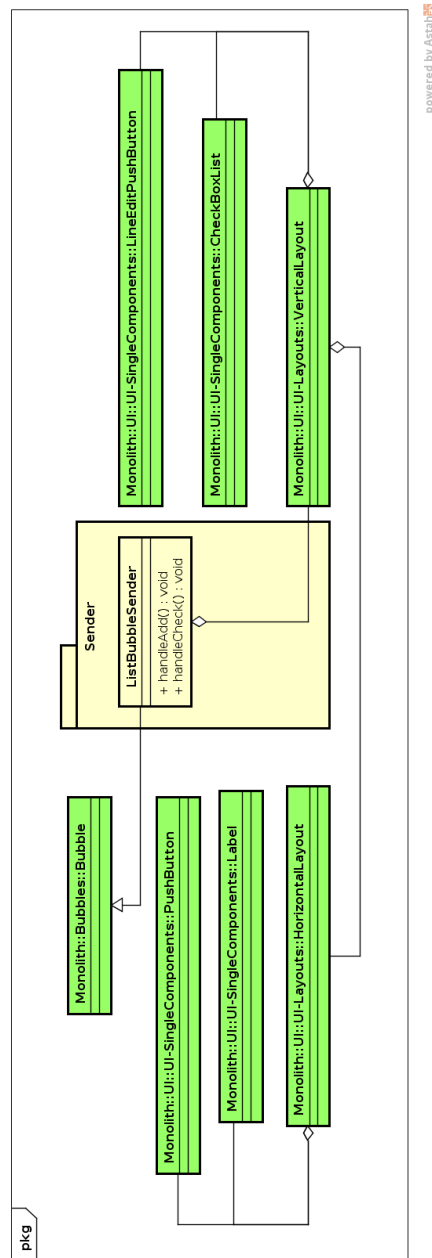


Figura 15: Diagramma per ListBubble::Sender.

Descrizione:

Componente che gestisce la visualizzazione della bolla da parte del mittente.

Classi contenute:

- ListBubbleSender

3.3.10 MeteoBubble

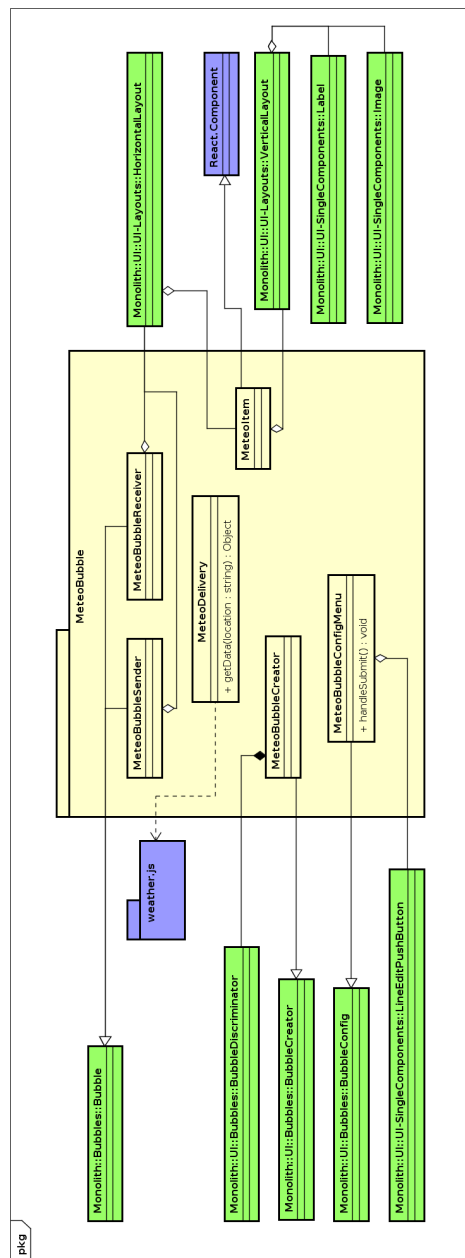


Figura 16: Diagramma per MeteoBubble.

Descrizione:

Componente contenente le classi necessarie per la creazione della bolla meteo.

Classi contenute:

- MeteoBubbleConfigMenu
- MeteoBubbleCreator



- MeteoBubbleReceiver
- MeteoBubbleSender
- MeteoDelivery
- MeteoItem



3.3.11 SurveyBubble

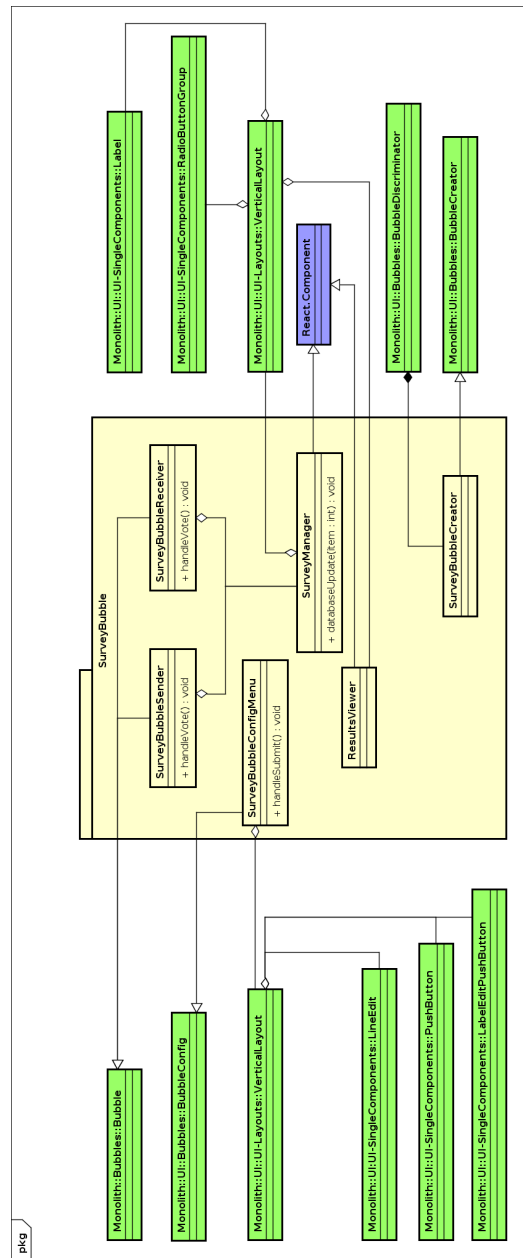


Figura 17: Diagramma per SurveyBubble.

Descrizione:

Componente contenente le classi necessarie per la creazione della bolla sondaggio.

Classi contenute:

- ResultsViewer
- SurveyBubbleConfigMenu



- SurveyBubbleCreator
- SurveyBubbleReceiver
- SurveyBubbleSender
- SurveyManager

3.3.12 TranslationBubble

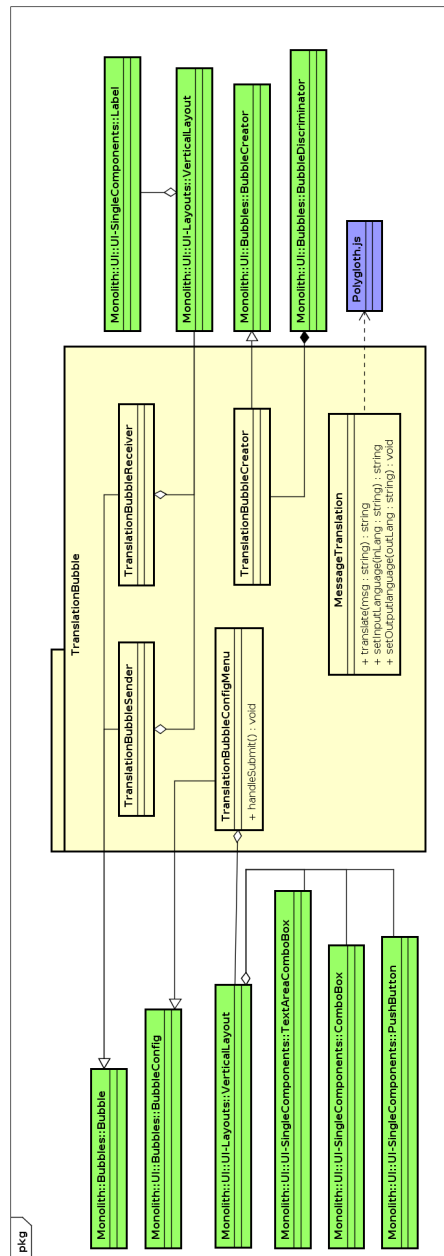


Figura 18: Diagramma per TranslationBubble.

Descrizione:

Componente contenente le classi necessarie per la creazione della bolla traduttore.

Classi contenute:

- MessageTranslation
- TranslationBubbleConfigMenu



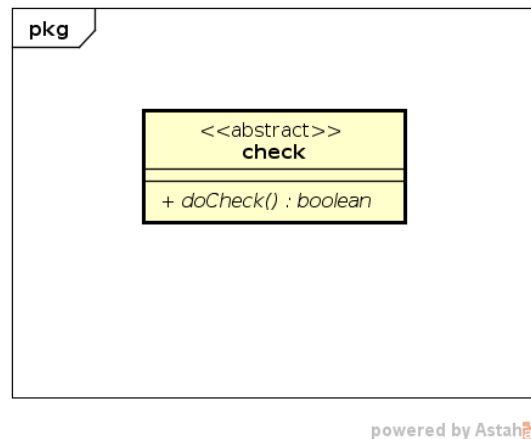
- TranslationBubbleCreator
- TranslationBubbleReceiver
- TranslationBubbleSender



3.4 Architettura di dettaglio - Classi del sistema Monolith

3.4.1 check

ComponenteMonolith::Database::informationStorage::Checks



powered by Astah

Figura 19: Diagramma per check in Checks

Descrizione

Classe concreta di concreteCheck che serve per effettuare controlli sui dati.

Metodi:

- +doCheck() : boolean
Ritorna il risultato di un controllo, true se positivo false se negativo.

Applicazioni

Interfaccia che viene utilizzata come rappresentazione di concreteCheck.



3.4.2 checkCreator

ComponenteMonolith::Database::informationStorage::Checks

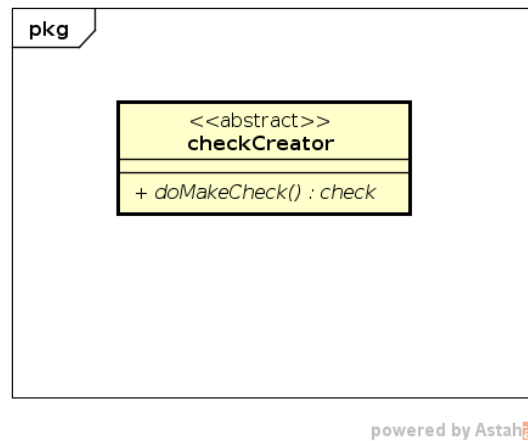


Figura 20: Diagramma per checkCreator in Checks

Descrizione

Classe astratta di concreteCheckCreator. **Metodi:**

- +doMakeCheck() : check
Ritorna un'istanza di check.

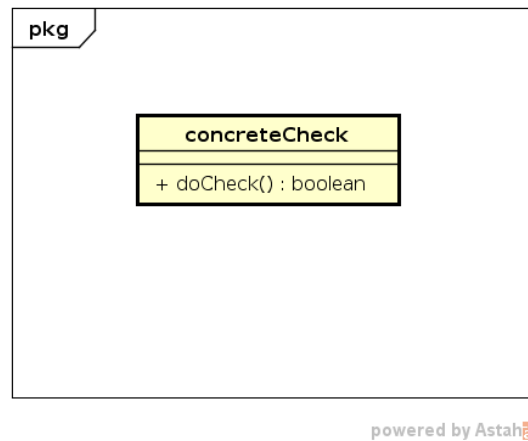
Applicazioni

Viene utilizzata quando viene richiesto un controllo.



3.4.3 concreteCheck

ComponenteMonolith::Database::informationStorage::Checks



powered by Astah

Figura 21: Diagramma per concreteCheck in Checks

Descrizione

Classe che effettua un controllo e ne ritorna il risultato. **Metodi:**

- +doCheck() : bool
Ritorna il risultato di un controllo, true se positivo false se negativo.

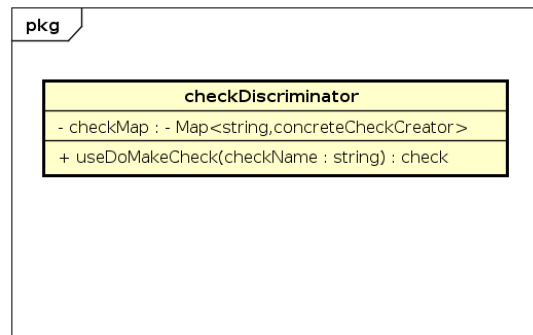
Applicazioni

Viene utilizzata per effettuare controlli sui dati.



3.4.4 checkDiscriminator

ComponenteMonolith::Database::informationStorage::Checks



powered by Astah

Figura 22: Diagramma per checkDiscriminator in Checks

Descrizione

Classe che effettua il controllo in base alla stringa considerata. **Attributi:**

- -checkMap : Map< string , concreteCheckCreator >
Struttura che mappa il nome del check con l'istanza del check.

Metodi:

- +useDoMakeCheck(checkName : string) : bool
Ritorna il risultato del controllo corrispondete alla stringa passata.

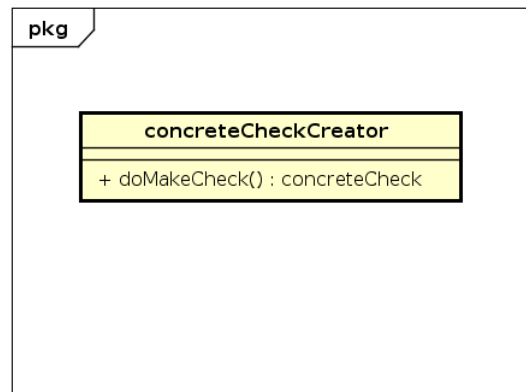
Applicazioni

Viene utilizzata quando viene richiesto un controllo.



3.4.5 concreteCheckCreator

ComponenteMonolith::Database::informationStorage::Checks



powered by Astah

Figura 23: Diagramma per concreteCheckCreator in Checks

Descrizione

Classe che rappresenta istanze concrete di tipo check. **Metodi:**

- +doMakeCheck() : concreteCheck
Ritorna un'istanza di concreteCheck.

Applicazioni

Viene utilizzata per creare istanze di controlli.



3.4.6 Bubble

ComponenteMonolith::UI::Bubbles

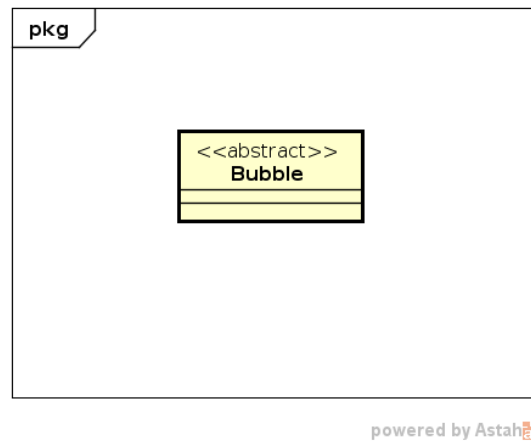


Figura 24: Diagramma per Bubble in Bubbles

Descrizione

Classe Astratta di ConcreteBubble.

Applicazioni

Viene utilizzata come interfaccia generica di bolla.



3.4.7 BubbleConfig

ComponenteMonolith::UI::Bubbles

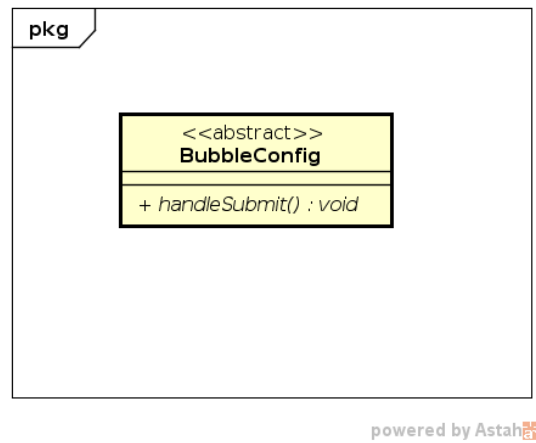


Figura 25: Diagramma per BubbleConfig in Bubbles

Descrizione

Classe Astratta di ConcreteBubbleConfig.

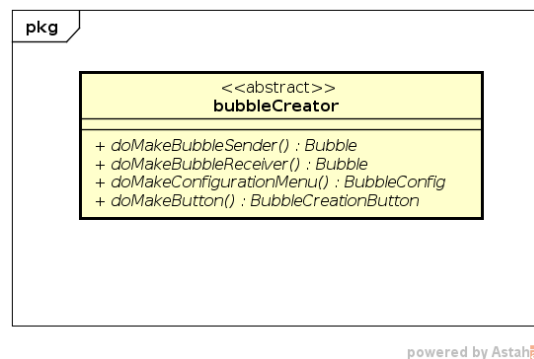
Applicazioni

Interfaccia che viene utilizzata come rappresentazione di ConcreteBubbleConfig.



3.4.8 bubbleCreator

ComponenteMonolith::UI::Bubbles



powered by Astah

Figura 26: Diagramma per bubbleCreator in Bubbles

Descrizione

Classe astratta di concreteBubbleCreator. **Metodi:**

- **+doMakeBubbleSender() : ConcreteBubble**
Ritorna il componente React per visualizzare una bolla inviata.
- **+doMakeBubbleReceiver() : ConcreteBubble**
Ritorna il componente React per visualizzare una bolla ricevuta.
- **+doMakeConfigurationMenu() : ConcreteBubbleConfig**
Ritorna il componente React per visualizzare il menù di creazione di una bolla da inviare.
- **+doMakeButton() : ConcreteBubbleCreationButton**
Ritorna il componente React per visualizzare il bottone di creazione del menù di configurazione di una bolla da inviare.

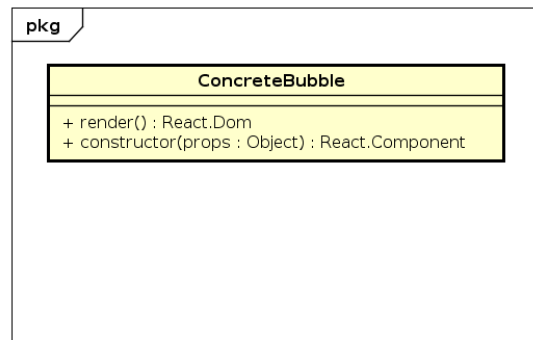
Applicazioni

Interfaccia che viene utilizzata come rappresentazione di concreteBubbleCreator.



3.4.9 ConcreteBubble

ComponenteMonolith::UI::Bubbles



powered by Astah

Figura 27: Diagramma per ConcreteBubble in Bubbles

Descrizione

Classe React che rappresenta una bolla inviata o ricevuta.

Metodi:

- `+constructor(props : Object) : React.Component`
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- `+render() : React.DOM`
Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

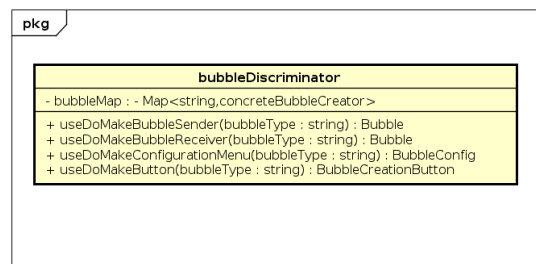
Applicazioni

Viene utilizzata da `concreteBubbleCreator` per creare una bolla inviata o ricevuto dai dati recuperati da `meteor`.



3.4.10 bubbleDiscriminator

ComponenteMonolith::UI::Bubbles



powered by Astah

Figura 28: Diagramma per bubbleDiscriminator in Bubbles

Descrizione

classe che contiene i metodi che ritornano le funzionalità necessarie per la rappresentazione della delle bolle.

Attributi:

- -bubbleMap : Map< string,concreteBubbleCreator>
Struttura che mappa il nome di una bolla con l'istanza di concreteBubbleCreator per quella bolla.

Metodi:

- +useDoMakeBubbleSender(bubbleType: string) : ConcreteBubble
Ritorna il componente React della stringa passata come parametro per visualizzare una bolla inviata.
- +useDoMakeBubbleReceiver(bubbleType: string) : ConcreteBubble
Ritorna il componente React della stringa passata come parametro per visualizzare una bolla ricevuta.
- +useDoMakeBubbleConfigurationMenu(bubbleType: string) : ConcreteBubbleConfig
Ritorna il componente React della stringa passata come parametro per visualizzare il menù di creazione di una bolla da inviare.
- +useDoMakeButton(bubbleType: string) : ConcreteBubbleCreationButton
Ritorna il componente React della stringa passata come parametro per visualizzare il bottone di creazione del menù di configurazione di una bolla da inviare.

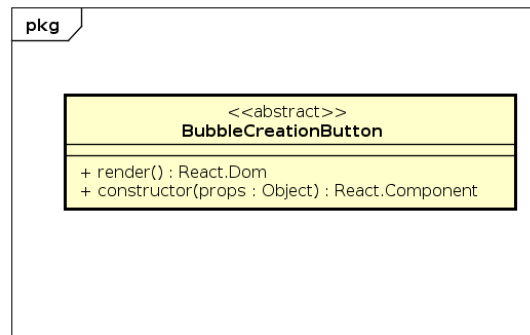
Applicazioni

Viene utilizzata quando si deve creare una nuova bolla, ritornando l'oggetto della bolla appena selezionata.



3.4.11 BubbleCreationButton

ComponenteMonolith::UI::Bubbles



powered by Astah

Figura 29: Diagramma per BubbleCreationButton in Bubbles

Descrizione

Classe Astratta di ConcreteBubbleCreationButton.

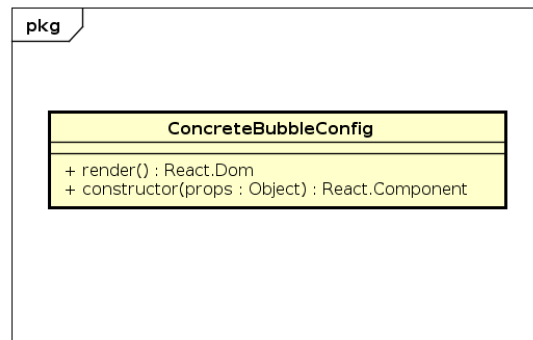
Applicazioni

Interfaccia che viene utilizzata come rappresentazione di ConcreteBubbleCreationButton.



3.4.12 ConcreteBubbleConfig

ComponenteMonolith::UI::Bubbles



powered by Astah

Figura 30: Diagramma per ConcreteBubbleConfig in Bubbles

Descrizione

Classe che rappresenta il menù di creazione di una bolla.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata da concreteBubbleCreator per creare un'istanza del menù di creazione di una bolla.



3.4.13 concreteBubbleCreator

ComponenteMonolith::UI::Bubbles

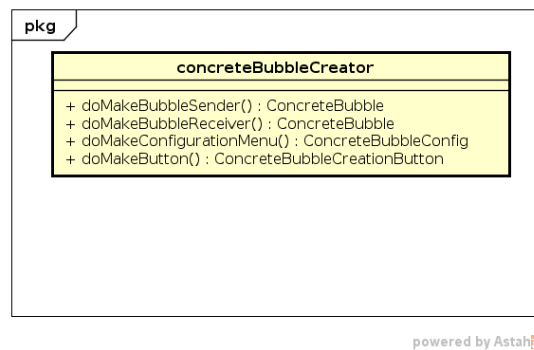


Figura 31: Diagramma per concreteBubbleCreator in Bubbles

Descrizione

Classe che contiene i factory method utilizzati per la creazione degli oggetti concreti di tipo bubble, BubbleConfig e BubbleCreationButton.

Metodi:

- +doMakeBubbleSender() : ConcreteBubble
Ritorna il componente React per visualizzare una bolla inviata.
- +doMakeBubbleReceiver() : ConcreteBubble
Ritorna il componente React per visualizzare una bolla ricevuta.
- +doMakeConfigurationMenu() : ConcreteBubbleConfig
Ritorna il componente React per visualizzare il menù di creazione di una bolla da inviare.
- +doMakeButton() : ConcreteBubbleCreationButton
Ritorna il componente React per visualizzare il bottone di creazione del menù di configurazione di una bolla da inviare.

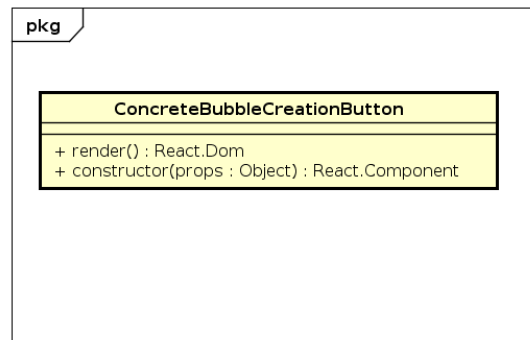
Applicazioni

Viene utilizzata da bubbleleDiscriminator per creare le bolle ricevute e inviate, i menù di configurazione delle bolle e i bottoni per aprire il menù di configurazione delle bolle.



3.4.14 ConcreteBubbleCreationButton

ComponenteMonolith::UI::Bubbles



powered by Astah

Figura 32: Diagramma per ConcreteBubbleCreationButton in Bubbles

Descrizione

Classe che rappresenta il bottone per aprire il menù di creazione di una bolla.

Metodi:

- `+constructor(props : Object) : React.Component`
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- `+render() : React.DOM`
Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento `React` che può essere una rappresentazione di un componente `DOM` nativo o un altro componente composito.

Applicazioni

Viene utilizzata da `concreteBubbleCreator` per creare un'istanza di un bottone nel menù di selezione di una bolla.



3.4.15 SideArea1

ComponenteMonolith::UI::SideAreas::SideArea1_pkg

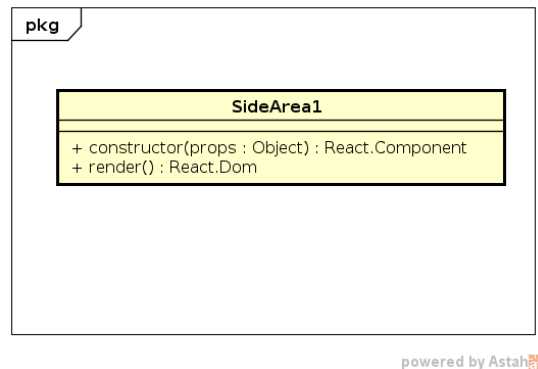


Figura 33: Diagramma per SideArea1 in SideArea1_pkg

Descrizione

Classe che rappresenta il componente React per la visualizzazione del contenuto della prima barra laterale. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata all'apertura della prima barra laterale per visualizzare il menù di creazione delle bolle e lo storico delle bolle inviate.



3.4.16 SentBubbleHistory

ComponenteMonolith::UI::SideAreas::SideArea1_pkg

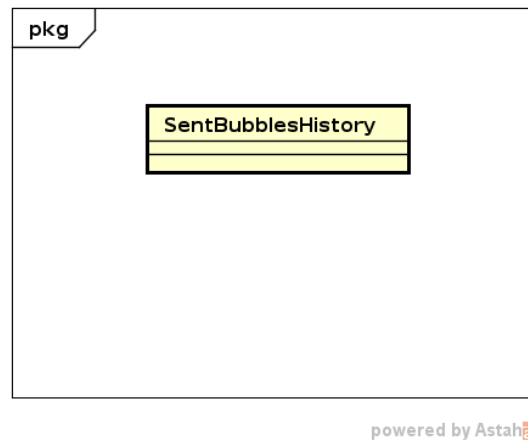


Figura 34: Diagramma per SentBubbleHistory in SideArea1_pkg

Descrizione

Classe che rappresenta il componente React per la visualizzazione dello storico delle bolle inviate.

Metodi:

- `+constructor(props : Object) : React.Component`
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- `+render() : React.DOM`
Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

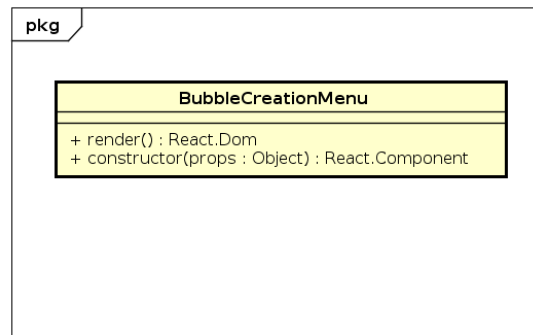
Applicazioni

Viene utilizzata dalla `SideArea1` per la visualizzazione dello storico delle bolle inviate.



3.4.17 BubbleCreationMenu

ComponenteMonolith::UI::SideAreas::SideArea1_pkg



powered by Astah

Figura 35: Diagramma per BubbleCreationMenu in SideArea1_pkg

Descrizione

Classe che rappresenta il componente React per la configurazione delle bolle.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

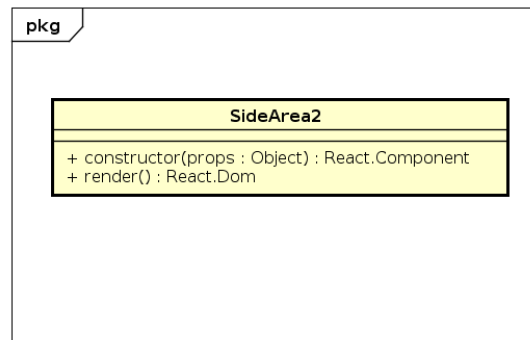
Applicazioni

Viene utilizzata dalla SideArea1 per la visualizzazione del menù di creazione delle bolle.



3.4.18 SideArea2

ComponenteMonolith::UI::SideAreas::SideArea2_pkg



powered by Astah

Figura 36: Diagramma per SideArea2 in SideArea2_pkg

Descrizione

Classe che rappresenta il componente React per la visualizzazione del contenuto della seconda barra laterale. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

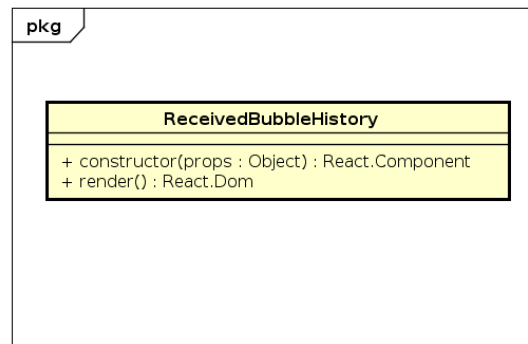
Applicazioni

Viene utilizzata all'apertura della seconda barra laterale per visualizzare dello storico delle bolle ricevute.



3.4.19 ReceivedBubbleHistory

ComponenteMonolith::UI::SideAreas::SideArea2_pkg



powered by Astah

Figura 37: Diagramma per ReceivedBubbleHistory in SideArea2_pkg

Descrizione

Classe che rappresenta il componente React per la visualizzazione dello storico delle bolle ricevute. **Metodi:**

- +constructor(props : Object) : React.Component
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- +render() : React.DOM
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

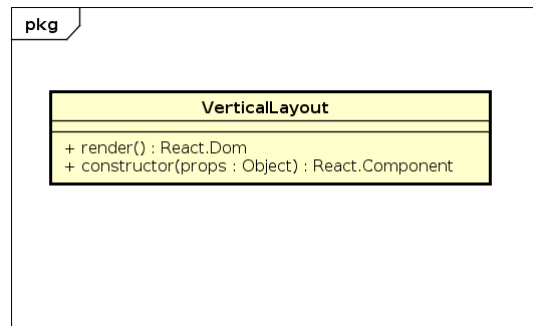
Applicazioni

Viene utilizzata dalla SideArea2 per la visualizzazione dello storico delle bolle ricevuto.



3.4.20 VerticalLayout

ComponenteMonolith::UI::UI-Layouts



powered by Astah

Figura 38: Diagramma per VerticalLayout in UI-Layouts

Descrizione

Classe contenitore che dispone i componenti contenuti uno sotto l'altro **Metodi:**

- +constructor(props : Object) : React.Component
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- +render() : React.DOM
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

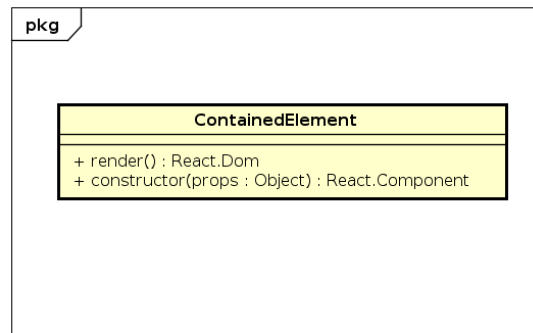
Applicazioni

Utilizzata per assegnare l'attributo className bootstrap a tutti i componenti figli, in modo che vengano visualizzati allineati in verticale con la dimensione adeguata.



3.4.21 ContainedElement

ComponenteMonolith::UI::UI-Layouts



powered by Astah

Figura 39: Diagramma per ContainedElement in UI-Layouts

Descrizione

Classe che rappresenta un oggetto figlio dei Layout **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

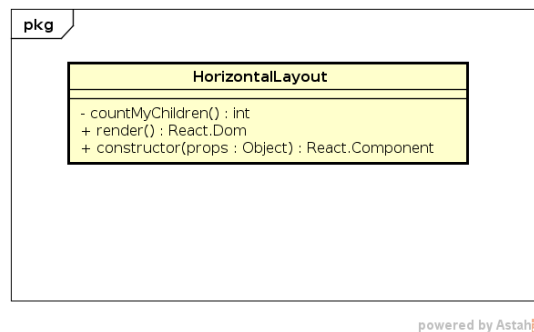
Applicazioni

Viene utilizzata dalle classi HorizontalLayout, VerticalLayout e ConditionalRendering per rappresentare un oggetto generico.



3.4.22 HorizontalLayout

ComponenteMonolith::UI::UI-Layouts



powered by Astah

Figura 40: Diagramma per HorizontalLayout in UI-Layouts

Descrizione

Classe contenitore che dispone i componenti contenuti in Horizontale. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.
- **+countMyChildren() : int**
Metodo che ritorna un intero contenente in numero di componenti figli, utilizzato per impostare la classe bootstrap corretta.

Applicazioni

Utilizzata per assegnare l'attributo className bootstrap a tutti i componenti figli, in modo che vengano visualizzati allineati in orizzontale con la dimensione adeguata.



3.4.23 ConditionalRendering

ComponenteMonolith::UI::UI-Layouts

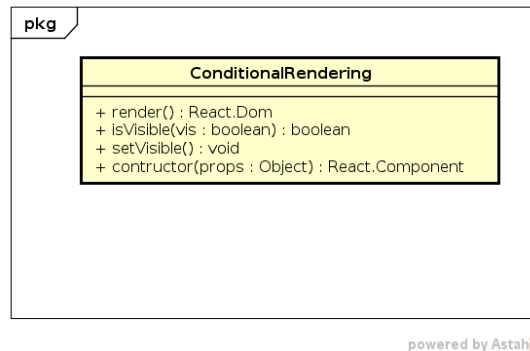


Figura 41: Diagramma per ConditionalRendering in UI-Layouts

Descrizione

Classe che fornisce i metodi per rendere un componente visibile o invisibile **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+setVisible(vis : boolean): void**
Imposta l'attributo che indica se il componente è visibile o meno in base al valore di vis.
- **+isVisible() : boolean**
Metodo che ritorna true se il componente è visibile altrimenti ritorna false.
- **+setVisible(vis : boolean) : void**
Imposta l'attributo che indica se il componente è visibile o meno in base al valore del parametro.
- **+isVisible() : bool**
Metodo che ritorna true se il componente è visibile altrimenti ritorna false.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata per la visibilità di un componente.



3.4.24 Image

ComponenteMonolith::UI::UISingleComponents

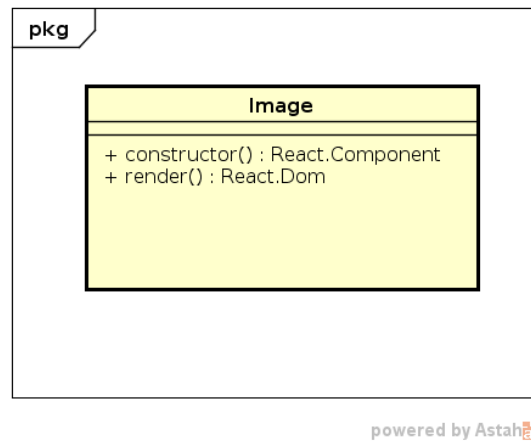


Figura 42: Diagramma per Image in UI-SingleComponents

Descrizione

Componente React che rappresenta un'immagine. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

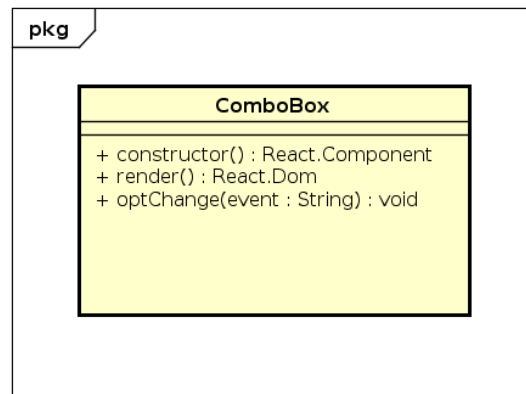
Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.25 ComboBox

ComponenteMonolith::UI::UISingleComponents



powered by Astah

Figura 43: Diagramma per ComboBox in UI-SingleComponents

Descrizione

Componente React che rappresenta un combobox. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+optChange(event : String) : void**
Cambia lo state del componente tenendo traccia dell'elemento selezionato e viene invocato il metodo del genitore per trasferirvi il cambiamento di stato
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.26 LineEdit

ComponenteMonolith::UI::UISingleComponents

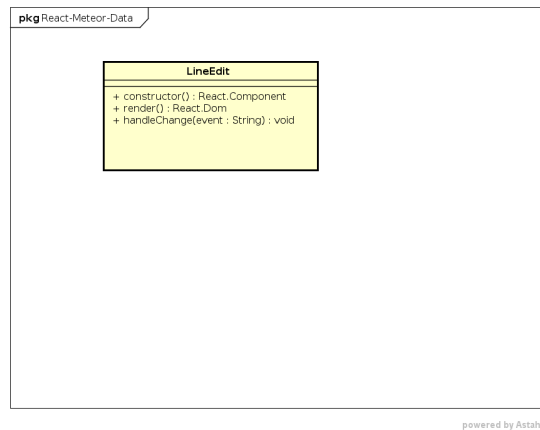


Figura 44: Diagramma per LineEdit in UI-SingleComponents

Descrizione

Componente React che rappresenta un input di testo. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+handleChange(event : String) : void**
Viene invocato il metodo del genitore per trasferirvi il cambiamento di stato
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

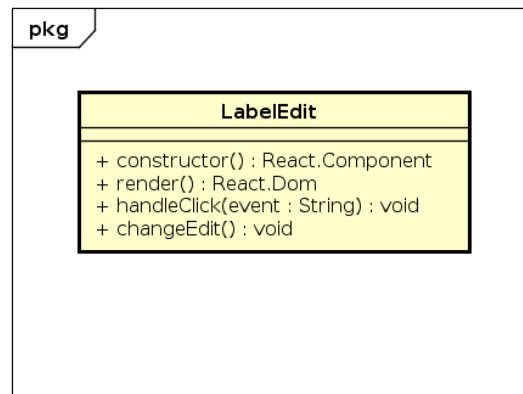
Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.27 LabelEdit

ComponenteMonolith::UI::UISingleComponents



powered by Astah

Figura 45: Diagramma per LabelEdit in UI-SingleComponents

Descrizione

Componente React che rappresenta un testo modificabile. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+handleClick(event : String) : void**
Viene invocato il metodo del genitore per trasferirvi il cambiamento di stato
- **+changeEdit() : void**
Cambia lo state del componente rendendolo (o meno) modificabile
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

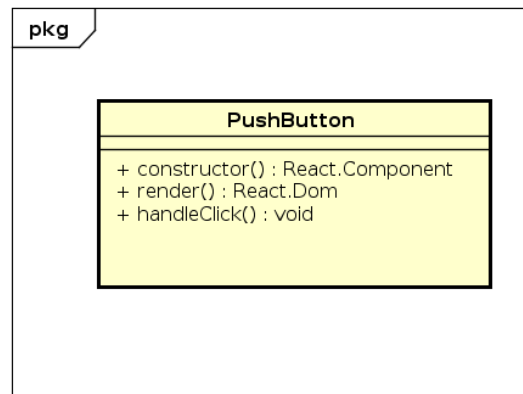
Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.28 PushButton

ComponenteMonolith::UI::UISingleComponents



powered by Astah

Figura 46: Diagramma per PushButton in UI-SingleComponents

Descrizione

Componente React che rappresenta un pulsante cliccabile. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+handleClick(): void**
Viene invocato il metodo passato dal genitore
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

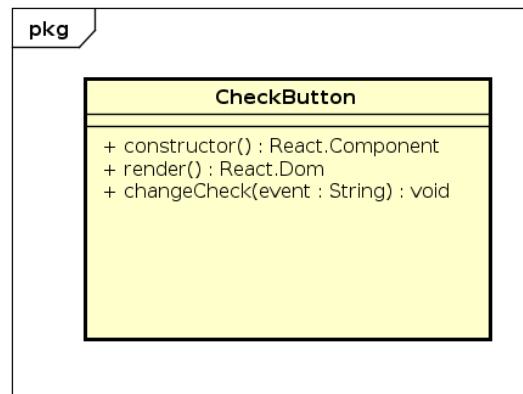
Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.29 CheckButton

ComponenteMonolith::UI::UI-SingleComponents



powered by Astah

Figura 47: Diagramma per CheckButton in UI-SingleComponents

Descrizione

Componente React che rappresenta un elemento di checkbox. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+changeCheck(event : String) : void**
Cambia lo state del componente tenendo traccia se l'elemento viene selezionato
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

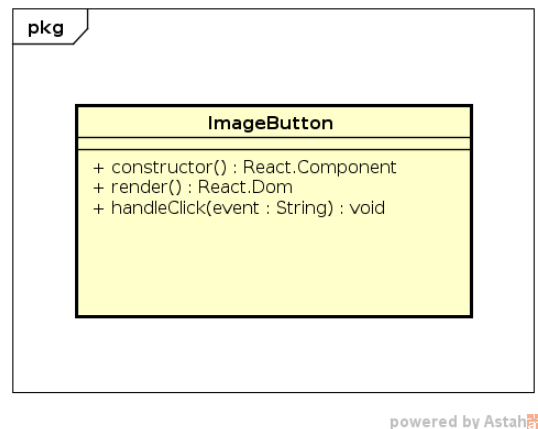
Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.30 ImageButton

ComponenteMonolith::UI::UISingleComponents



powered by Astah

Figura 48: Diagramma per ImageButton in UI-SingleComponents

Descrizione

Componente React che rappresenta un immagine che funge da pulsante. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+handleClick(event : String) : void**
Viene invocato il metodo passato dal genitore.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

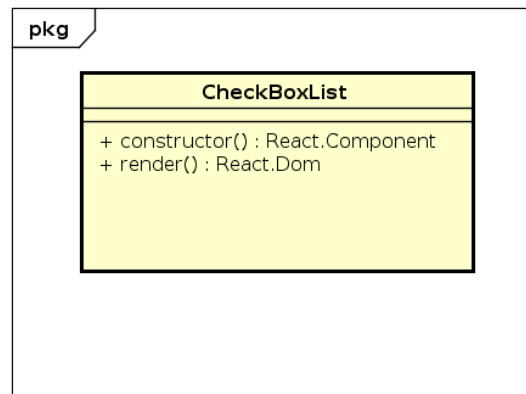
Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.31 CheckBoxList

ComponenteMonolith::UI::UI-SingleComponents



powered by Astah

Figura 49: Diagramma per CheckBoxList in UI-SingleComponents

Descrizione

Componente React che rappresenta una lista di CheckBox. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

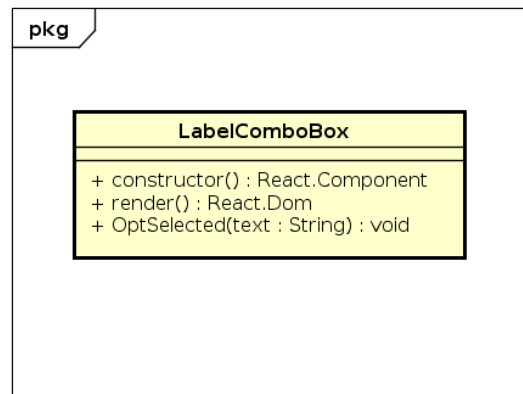
Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.32 LabelComboBox

ComponenteMonolith::UI::UISingleComponents



powered by Astah

Figura 50: Diagramma per LabelComboBox in UI-SingleComponents

Descrizione

Componente React che rappresenta una lista di item selezionabile affiancata da una Label. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+OptSelected(text : String) : void**
Modifica lo state tenendo traccia del item selezionato.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

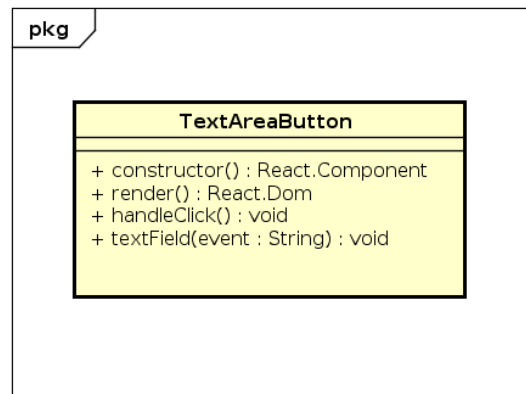
Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.33 TextAreaButton

ComponenteMonolith::UI::UISingleComponents



powered by Astah

Figura 51: Diagramma per TextAreaButton in UI-SingleComponents

Descrizione

Componente React che rappresenta un'area di testo con un pulsante. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+textField(text : String) : void**
Modifica lo state tenendo traccia del testo inserito nel Label
- **+handleClick(event : String) : void**
Viene invocato il metodo passato dal genitore
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.34 LabelPushButton

ComponenteMonolith::UI::UISingleComponents

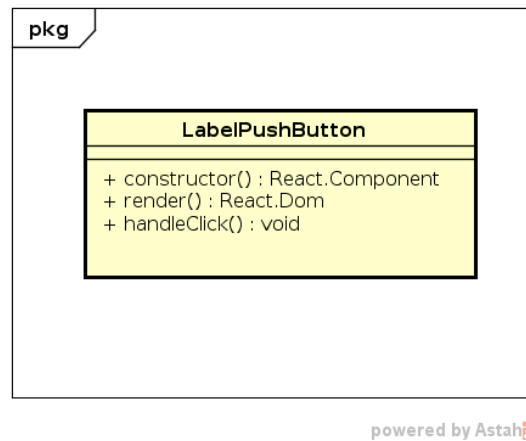


Figura 52: Diagramma per LabelPushButton in UI-SingleComponents

Descrizione

Componente React che rappresenta un Label affiancato da un bottone cliccabile.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+handleClick(event : String) : void**
Viene invocato il metodo passato dal genitore
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.35 LineEditComboBox

ComponenteMonolith::UI::UISingleComponents

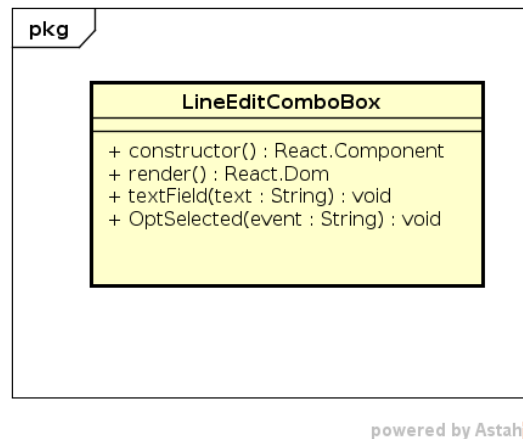


Figura 53: Diagramma per LineEditComboBox in UI-SingleComponents

Descrizione

Componente React che rappresenta una lista di aree di testo selezionabile.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+textField(text : String) : void**
Modifica lo state tenendo traccia del testo inserito in ciascuna area.
- **+OptSelected(event : String) : void**
Viene invocato il metodo passato dal genitore
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

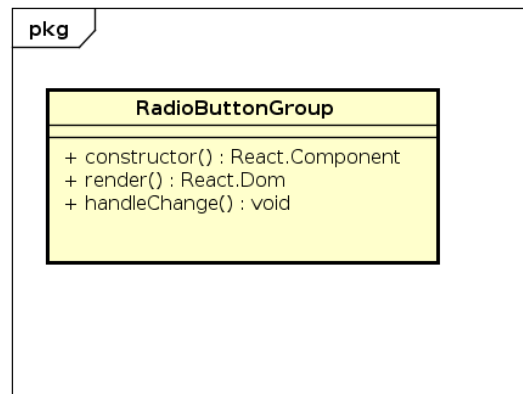
Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.36 RadioButtonGroup

ComponenteMonolith::UI::UISingleComponents



powered by Astah

Figura 54: Diagramma per RadioButtonGroup in UI-SingleComponents

Descrizione

Componente React che rappresenta un'area di testo con un pulsante. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+handleChange(): void**
Viene invocato il metodo passato dal genitore
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

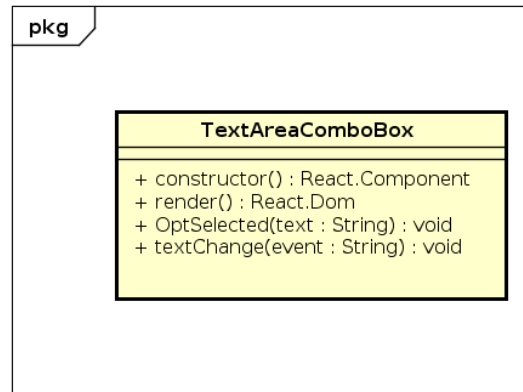
Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.37 TextAreaComboBox

ComponenteMonolith::UI::UISingleComponents



powered by Astah

Figura 55: Diagramma per TextAreaComboBox in UI-SingleComponents

Descrizione

Componente React che rappresenta una lista di TextArea selezionabile. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+OptSelected(text : String) : void**
Modifica lo state 'selected' tenendo traccia del TextArea selezionata ed aggiorna il metodo invocato dal genitore.
- **+textChange(event : String) : void**
Modifica lo state 'value' tenendo traccia del testo digitato nel TextAre ed aggiorna il metodo invocato dal genitore.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

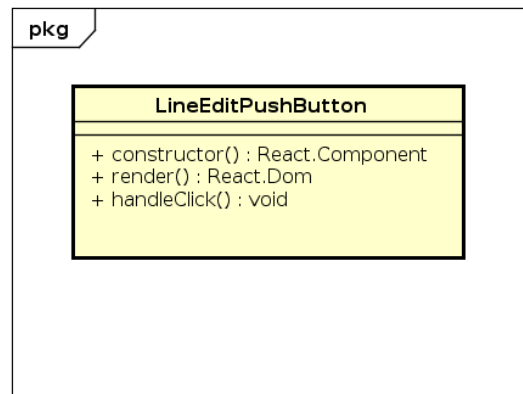
Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.38 LineEditPushButton

ComponenteMonolith::UI::UISingleComponents



powered by Astah

Figura 56: Diagramma per LineEditPushButton in UI-SingleComponents

Descrizione

Componente React che rappresenta un combobox. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+handleClick(event : String) : void**
Viene invocato il metodo passato dal genitore
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

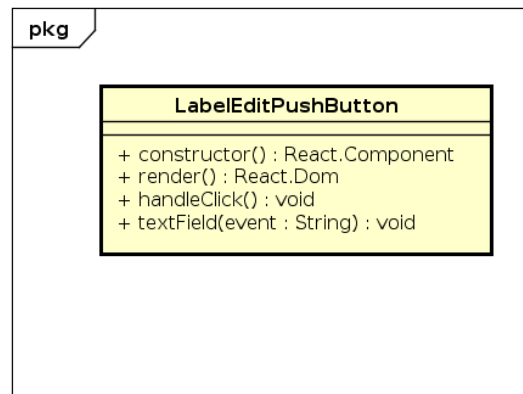
Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.



3.4.39 LabelEditPushButton

ComponenteMonolith::UI::UISingleComponents



powered by Astah

Figura 57: Diagramma per LabelEditPushButton in UI-SingleComponents

Descrizione

Componente React che rappresenta del testo con un pulsante. **Metodi:**

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+textField(text : String) : void**
Modifica lo state tenendo traccia del testo inserito nel Label
- **+handleClick(event : String) : void**
Viene invocato il metodo passato dal genitore
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzato per costruire le interfacce grafiche delle bolle.

3.5 Architettura di dettaglio - Classi delle bolle demo

3.5.1 CurrencyConversion

ComponenteCurrencyBubble

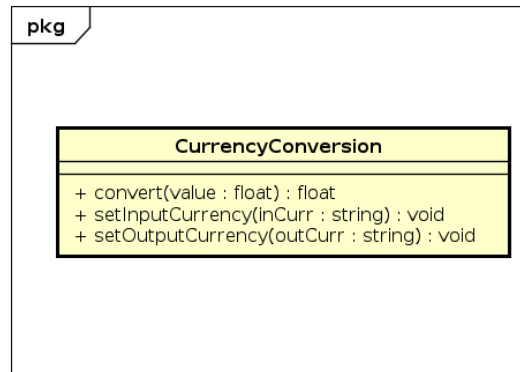


Figura 58: Diagramma per CurrencyConversion in

Descrizione

Classe predisposta per la conversione di un importo da una valuta di partenza a una valuta di arrivo.

Metodi:

- **+convert(value : float) : float**
Metodo che effettua la conversione del valore utilizzando l'utility money.js.
- **+setInputCurrency(inCurr : string) : void**
Metodo che imposta la valuta di partenza.
- **+setOutputCurrency(outCurr : string) : void**
Metodo che imposta la valuta di arrivo.

Applicazioni

Viene utilizzata per effettuare la conversione di un importo dalla valuta di partenza a quella di arrivo. Si serve dell'utility money.js.



3.5.2 CurrencyBubbleSender

ComponenteCurrencyBubble

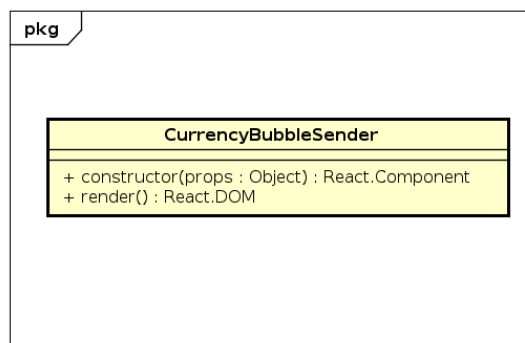


Figura 59: Diagramma per CurrencyBubbleSender in

Descrizione

Istanziatura concreta della classe Monolith::Bubbles::Bubble che rappresenta la bolla vista dal mittente.

Metodi:

- +constructor(props : Object) : React.Component
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- +render() : React.DOM Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata per la visualizzazione della bolla da parte del mittente. La classe è composta da Monolith::UI::UI-Layouts::HorizontalLayout (composta a sua volta da Monolith::UI::UI-SingleComponents::Label) che dispone i componenti visuali uno accanto all'altro.



3.5.3 CurrencyBubbleCreator

ComponenteCurrencyBubble

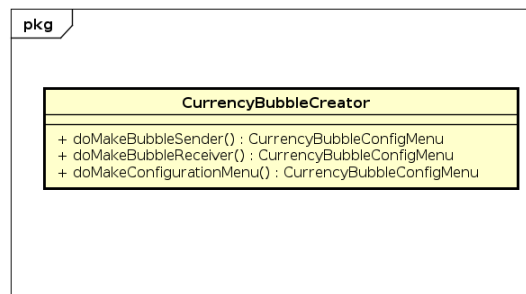


Figura 60: Diagramma per CurrencyBubbleCreator in

Descrizione

Istanziatura concreta della classe Monolith::UI::Bubbles::BubbleCreator che gestisce la creazione della specifica bolla convertitore di valuta, della specifica area di configurazione della bolla e dello specifico pulsante tramite l'utilizzo della classe Monolith::UI::Bubbles::BubbleDiscriminator.

Metodi:

- +doMakeBubbleSender() : CurrencyBubble
Metodo che gestisce la creazione della bolla vista dal mittente.
- +doMakeBubbleReceiver() : CurrencyBubble
Metodo che gestisce la creazione della bolla vista dal ricevente.
- +doMakeConfigurationMenu() : CurrencyBubbleConfig
Metodo che gestisce la creazione dell'area di configurazione della bolla.
- +doMakeButton() : CurrencyBubbleCreationButton
Metodo che gestisce la creazione dello specifico pulsante da inserire nel menu iniziale di creazione. Viene lasciata l'implementazione della super classe.

Applicazioni

Viene utilizzata per gestire la creazione della specifica bolla convertitore di valuta, della specifica area di configurazione della bolla e dello specifico pulsante.



3.5.4 CurrencyBubbleReceiver

ComponenteCurrencyBubble

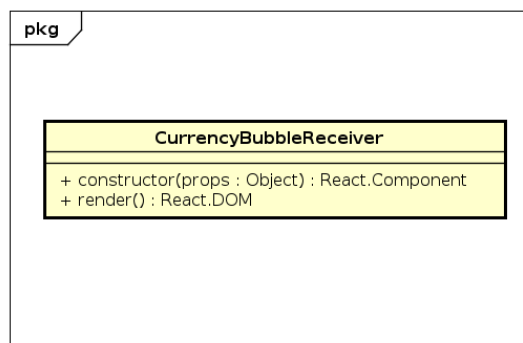


Figura 61: Diagramma per CurrencyBubbleReceiver in

Descrizione

Istanziatura concreta della classe Monolith::Bubbles::Bubble che rappresenta la bolla vista dal ricevente.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM** Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata per la visualizzazione della bolla da parte del ricevente. La classe è composta da Monolith::UI::UI-Layouts::HorizontalLayout (composta a sua volta da Monolith::UI::UI-SingleComponents::Label) che dispone i componenti visuali uno accanto all'altro.



3.5.5 CurrencyBubbleConfigMenu

ComponenteCurrencyBubble

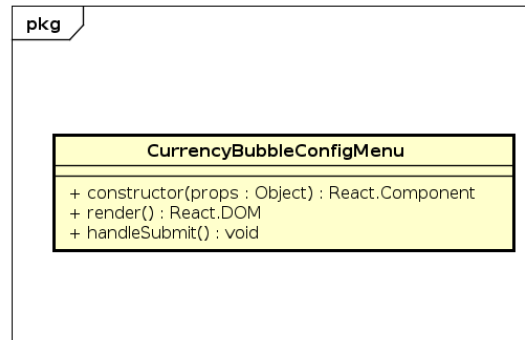


Figura 62: Diagramma per CurrencyBubbleConfigMenu in

Descrizione

Istanziatura concreta della classe `Monolith::UI::Bubbles::BubbleConfig` che rappresenta l'area di configurazione della bolla.

Metodi:

- `+constructor(props : Object) : React.Component`
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- `+render() : React.DOM` Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento `React` che può essere una rappresentazione di un componente `DOM` nativo o un altro componente composito.
- `+ handleSubmit() : void` Metodo che viene invocato quando la configurazione della bolla è terminata ed è pronta per essere inviata.

Applicazioni

Viene utilizzata per la visualizzazione dell'area di configurazione della bolla. La classe è composta da `Monolith::UI::UI-Layouts::VerticalLayout` (composta a sua volta da `Monolith::UI::UI-SingleComponents::LineEditComboBox` e `Monolith::UI::UI-SingleComponents::LabelComboBox`) che dispone i componenti visivi uno sotto l'altro.



3.5.6 DiceRoller

ComponenteDiceBubble

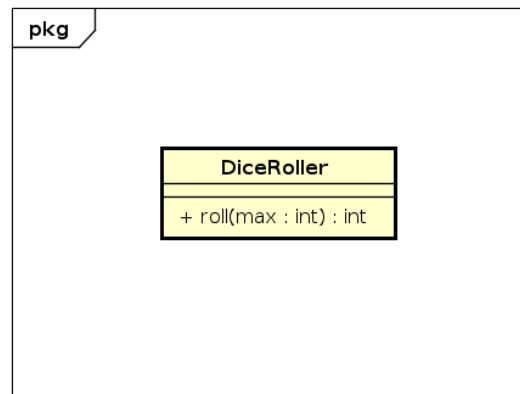


Figura 63: Diagramma per DiceRoller in

Descrizione

Classe predisposta per l'estrazione del numero casuale.

Metodi:

- **+roll(max : int) : int** Metodo che effettua l'estrazione del numero casuale utilizzando la funzione Javascript `Math.random()`.

Applicazioni

Viene utilizzata per l'estrazione del numero casuale compreso tra 0 e max. Si serve della funzione Javascript `Math.random()`.



3.5.7 DiceBubbleSender

ComponenteDiceBubble

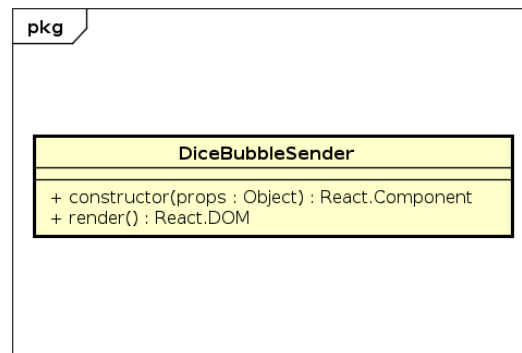


Figura 64: Diagramma per DiceBubbleSender in

Descrizione

Istanziatura concreta della classe Monolith::Bubbles::Bubble che rappresenta la bolla vista dal mittente.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM** Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata per la visualizzazione della bolla da parte del mittente. La classe è composta da Monolith::UI::UI-SingleComponents::Label.



3.5.8 DiceBubbleCreator

ComponenteDiceBubble

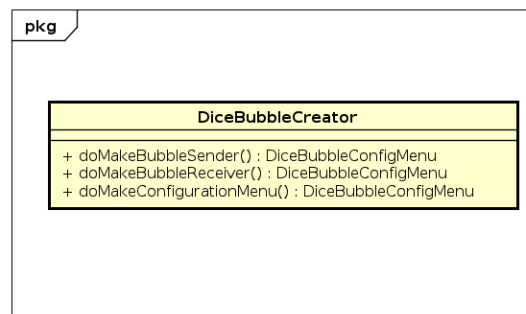


Figura 65: Diagramma per DiceBubbleCreator in

Descrizione

Istanziamento concreto della classe `Monolith::UI::Bubbles::BubbleCreator` che gestisce la creazione della specifica bolla estrazione numero casuale, della specifica area di configurazione della bolla e dello specifico pulsante tramite l'utilizzo della classe `Monolith::UI::Bubbles::BubbleDiscriminator`.

Metodi:

- `+doMakeBubbleSender() : DiceBubble`
Metodo che gestisce la creazione della bolla vista dal mittente.
- `+doMakeBubbleReceiver() : DiceBubble`
Metodo che gestisce la creazione della bolla vista dal ricevente.
- `+doMakeConfigurationMenu() : DiceBubbleConfig`
Metodo che gestisce la creazione dell'area di configurazione della bolla.
- `+doMakeButton() : DiceBubbleCreationButton`
Metodo che gestisce la creazione dello specifico pulsante da inserire nel menu iniziale di creazione. Viene lasciata l'implementazione della super classe.

Applicazioni

Viene utilizzata per gestire la creazione della specifica bolla estrazione numero casuale, della specifica area di configurazione della bolla e dello specifico pulsante.



3.5.9 DiceBubbleReceiver

ComponenteDiceBubble

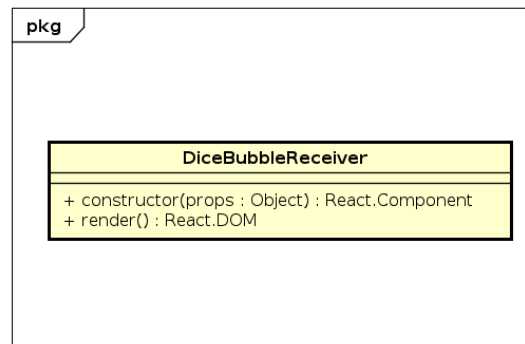


Figura 66: Diagramma per DiceBubbleReceiver in

Descrizione

Istanziatura concreta della classe Monolith::Bubbles::Bubble che rappresenta la bolla vista dal ricevente.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM** Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata per la visualizzazione della bolla da parte del ricevente. La classe è composta da Monolith::UI::UI-SingleComponents::Label.



3.5.10 DiceBubbleConfigMenu

ComponenteDiceBubble

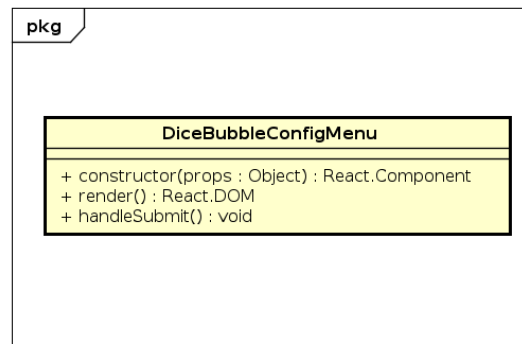


Figura 67: Diagramma per DiceBubbleConfigMenu in

Descrizione

Istanziatura concreta della classe `Monolith::UI::Bubbles::BubbleConfig` che rappresenta l'area di configurazione della bolla.

Metodi:

- `+constructor(props : Object) : React.Component`
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- `+render() : React.DOM` Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento `React` che può essere una rappresentazione di un componente `DOM` nativo o un altro componente composito.
- `+handleSubmit() : void` Metodo che viene invocato quando la configurazione della bolla è terminata ed è pronta per essere inviata.

Applicazioni

Viene utilizzata per la visualizzazione dell'area di configurazione della bolla. La classe è composta da `Monolith::UI::UI-SingleComponents::LineEditPushButton`.



3.5.11 CheckListCreator

ComponenteListBubble::CheckListCreation

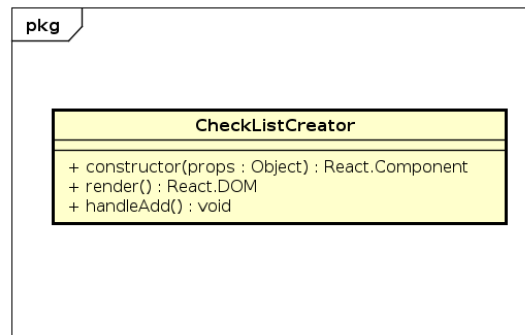


Figura 68: Diagramma per CheckListCreator in CheckListCreation

Descrizione

Componente React che rappresenta l'area di visualizzazione delle check list e un pulsante per poterne aggiungere di nuove.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM** Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.
- **+handleAdd() : void** Metodo che gestisce l'aggiunta di una nuova check list.

Applicazioni

Viene utilizzato per rappresentare l'area di visualizzazione delle check list e un pulsante per poterne aggiungere di nuove. La classe è composta da Monolith::UI::UI-Layouts::VerticalLayout (composta a sua volta da Monolith::UI::UI-SingleComponents::PushButton e CheckListComponent) che dispone i componenti visuali uno sotto l'altro.



3.5.12 CheckListComponent

ComponenteListBubble::CheckListCreation

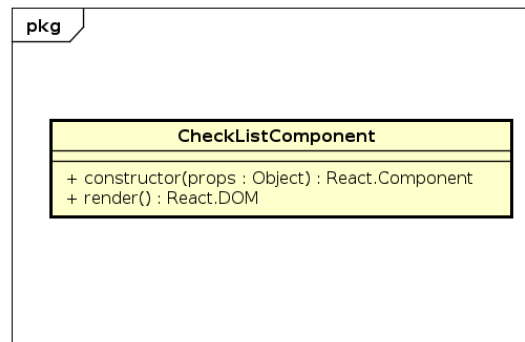


Figura 69: Diagramma per CheckListComponent in CheckListCreation

Descrizione

Componente React che rappresenta una lista di check list.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM** Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzato per rappresentare una lista di check list. La classe è composta da Monolith::UI::UI-SingleComponents::PushButton.



3.5.13 CheckListItemsDefinition

ComponenteListBubble::CheckListCreation

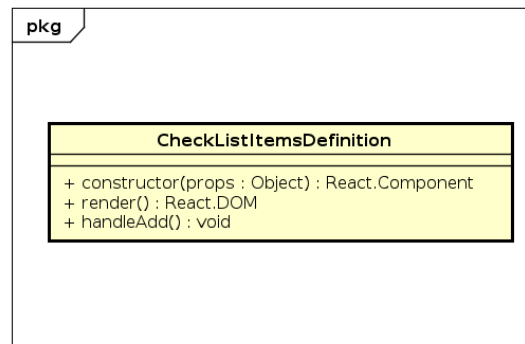


Figura 70: Diagramma per CheckListItemsDefinition in CheckListCreation

Descrizione

Componente React che rappresenta l'area di configurazione di una check list.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM** Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzato per rappresentare l'area di configurazione di una check list. La classe è composta da Monolith::UI::UI-Layouts::VerticalLayout (composta a sua volta da Monolith::UI::UI-SingleComponents::LineEdit e Monolith::UI::UI-SingleComponents::LineEditPushButton) che dispone i componenti visivi uno sotto l'altro.



3.5.14 CheckList

ComponenteListBubble::CheckListReading

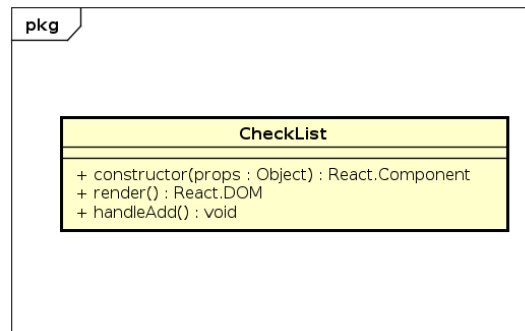


Figura 71: Diagramma per CheckList in CheckListReading

Descrizione

Componente React che gestisce l'aggiunta di un elemento alla lista.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.
- **+handleAdd() : void**
Metodo che gestisce l'aggiunta di un elemento alla lista.

Applicazioni

Viene utilizzata per gestire l'aggiunta di un elemento alla lista. La classe è composta da Monolith::UI::UI-Layouts::VerticalLayout (composta a sua volta da Monolith::UI::UI-SingleComponents::LabelEditPushButton) che dispone i componenti visivi uno sotto l'altro.



3.5.15 ListOfCheckLists

ComponenteListBubble::CheckListReading

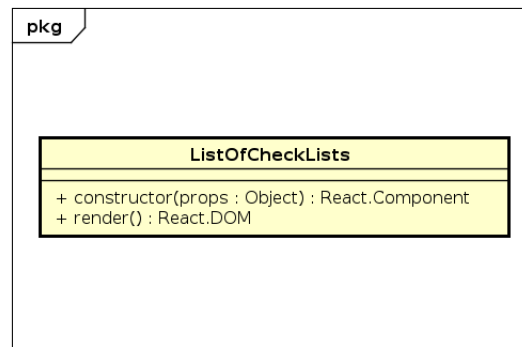


Figura 72: Diagramma per ListOfCheckLists in CheckListReading

Descrizione

Componente React che rappresenta una lista di check list.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM** Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata per la visualizzazione della lista di check list. La classe è composta da Monolith::UI::UI-Layouts::VerticalLayout (composta a sua volta da Monolith::UI::UI-SingleComponents::PushButton) che dispone gli elementi uno sotto l'altro.



3.5.16 ListCreationButton

ComponenteListBubble::Configuration

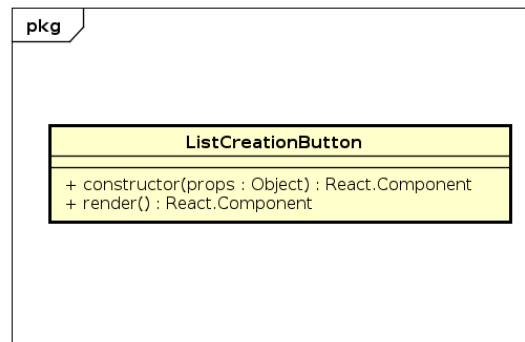


Figura 73: Diagramma per ListCreationButton in Configuration

Descrizione

Istanziamento della classe Monolith::UI::Bubbles::BubbleCreationButton che rappresenta lo specifico pulsante da inserire nel menu iniziale di creazione.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM** Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata per rappresentare lo specifico pulsante da inserire nel menu iniziale di creazione.



3.5.17 ListBubbleMenuConfig

ComponenteListBubble::Configuration

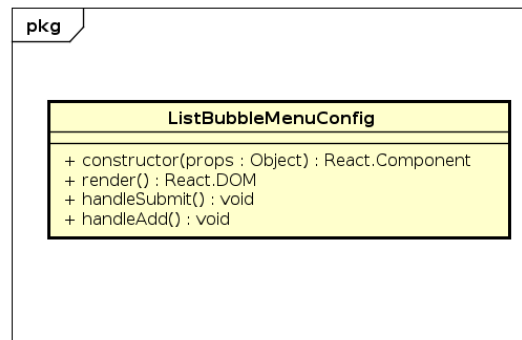


Figura 74: Diagramma per ListBubbleMenuConfig in Configuration

Descrizione

Istanziatura concreta della classe `Monolith::UI::Bubbles::BubbleConfig` che rappresenta l'area di configurazione della bolla.

Metodi:

- `+constructor(props : Object) : React.Component`
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- `+render() : React.DOM`
Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento `React` che può essere una rappresentazione di un componente `DOM` nativo o un altro componente composito.
- `+handleSubmit() : void`
Metodo che viene invocato quando la configurazione della bolla è terminata ed è pronta per essere inviata.
- `+handleAdd() : void`
Metodo che viene invocato quando si vuole aggiungere un elemento alla lista.

Applicazioni

Viene utilizzata per la visualizzazione dell'area di configurazione della bolla. La classe è composta da `Monolith::UI::UI-Layouts::VerticalLayout` (composta a sua volta da `Monolith::UI::UI-Layouts::HorizontalLayout` con un `LineEdit` e un `PushButton`, `Monolith::UI::UI-SingleComponents::LineEditPushButton`, `Monolith::UI::UI-SingleComponents::PushButton`) che dispone i componenti visivi uno sotto l'altro.



3.5.18 ListBubbleCreator

ComponenteListBubble::DataManagement

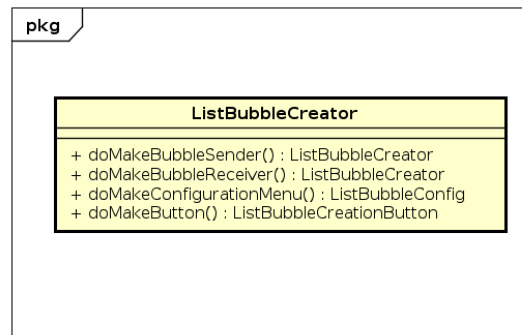


Figura 75: Diagramma per ListBubbleCreator in DataManagement

Descrizione

Istanziatura concreta della classe Monolith::UI::Bubbles::BubbleCreator che gestisce la creazione della specifica bolla lista, della specifica area di configurazione della bolla e dello specifico pulsante tramite l'utilizzo della classe Monolith::UI::Bubbles::BubbleDiscriminator.

Metodi:

- +doMakeBubbleSender() : ListBubble
Metodo che gestisce la creazione della bolla vista dal mittente.
- +doMakeBubbleReceiver() : ListBubble
Metodo che gestisce la creazione della bolla vista dal ricevente.
- +doMakeConfigurationMenu() : ListBubbleConfig
Metodo che gestisce la creazione dell'area di configurazione della bolla.
- +doMakeButton() : ListBubbleCreationButton
Metodo che gestisce la creazione dello specifico pulsante da inserire nel menu iniziale di creazione.

Applicazioni

Viene utilizzata per gestire la creazione della specifica bolla lista, della specifica area di configurazione della bolla e dello specifico pulsante.



3.5.19 ListBubbleReceiver

ComponenteListBubble::Receiver

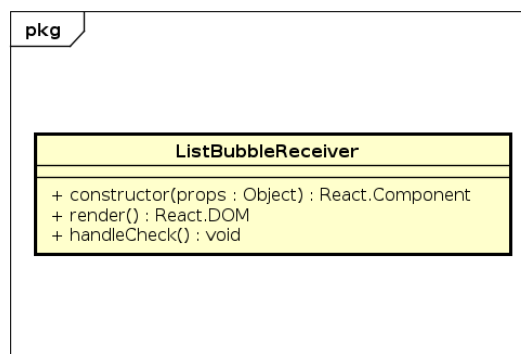


Figura 76: Diagramma per ListBubbleReceiver in Receiver

Descrizione

Istanziatura concreta della classe Monolith::Bubbles::Bubble che rappresenta la bolla vista dal ricevente.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.
- **+handleCheck() : void**
Metodo che gestisce la spunta di un elemento della lista.

Applicazioni

Viene utilizzata per la visualizzazione della bolla da parte del ricevente. La classe è composta da Monolith::UI::UI-Layouts::VerticalLayout (composta a sua volta da Monolith::UI::UI-SingleComponents::Label e Monolith::UI::UI-SingleComponents::CheckBoxList) che dispone i componenti visuali uno sotto l'altro.



3.5.20 ListBubbleSender

ComponenteListBubble::Sender

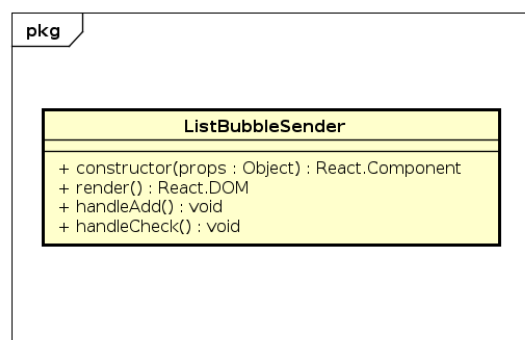


Figura 77: Diagramma per ListBubbleSender in Sender

Descrizione

Istanziatura concreta della classe Monolith::Bubbles::Bubble che rappresenta la bolla vista dal mittente.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM**
Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.
- **+handleAdd() : void**
Metodo che gestisce l'aggiunta di un nuovo elemento alla lista.
- **+handleCheck() : void**
Metodo che gestisce la spunta di un elemento della lista.

Applicazioni

Viene utilizzata per la visualizzazione della bolla da parte del mittente. La classe è composta da Monolith::UI::UI-Layouts::VerticalLayout (composta a sua volta da Monolith::UI::UI-SingleComponents::Label, Monolith::UI::UI-SingleComponents::PushButton, Monolith::UI::UI-SingleComponents::CheckBoxList, Monolith::UI::UI-SingleComponents::LineEditPushButton) che dispone i componenti visuali uno sotto l'altro.



3.5.21 MeteoItem

ComponenteMeteoBubble

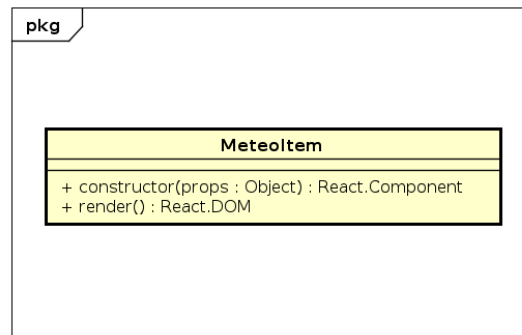


Figura 78: Diagramma per MeteoItem in

Descrizione

Componente React predisposto per la visualizzazione delle previsioni meteo di un determinato giorno.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM** Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata per la visualizzazione delle previsioni meteo di un determinato giorno. È composta da `Monolith::UI::UI-Layouts::VerticalLayout` (composta a sua volta da `Monolith::UI::UI-SingleComponents::Image` e `Monolith::UI::UI-SingleComponents::Label`) che dispone i componenti visuali uno sotto l'altro.



3.5.22 MeteoDelivery

ComponenteMeteoBubble

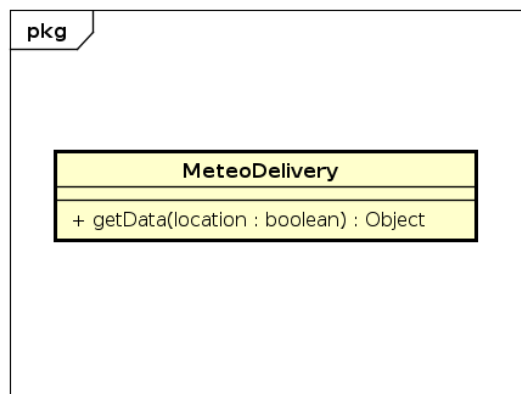


Figura 79: Diagramma per MeteoDelivery in

Descrizione

Classe predisposta per il recupero dei dati relativi alle previsioni meteo.

Metodi:

- +getData(location : string) : Object Metodo che recupera i dati relativi alle previsioni meteo della località selezionata utilizzando l'utility weather.js.

Applicazioni

Viene utilizzata per recuperare i dati relativi alle previsioni meteo della località scelta. Si serve dell'utility weather.js.



3.5.23 MeteoBubbleSender

ComponenteMeteoBubble

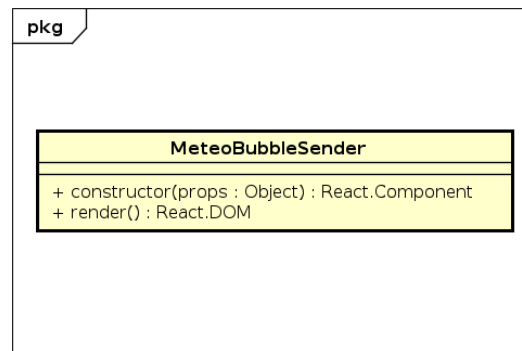


Figura 80: Diagramma per MeteoBubbleSender in

Descrizione

Istanziatura concreta della classe Monolith::Bubbles::Bubble che rappresenta la bolla vista dal mittente.

Metodi:

- +constructor(props : Object) : React.Component
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- +render() : React.DOM Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata per la visualizzazione della bolla da parte del mittente. La classe è composta da Monolith::UI::UI-Layouts::HorizontalLayout (composta a sua volta da MeteoItem) che dispone i componenti visuali uno accanto all'altro.



3.5.24 MeteoBubbleCreator

ComponenteMeteoBubble

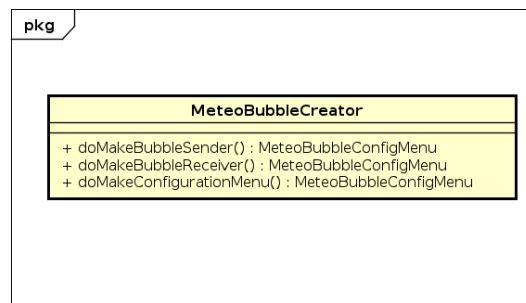


Figura 81: Diagramma per MeteoBubbleCreator in

Descrizione

Istanziamento concreto della classe `Monolith::UI::Bubbles::BubbleCreator` che gestisce la creazione della specifica bolla meteo, della specifica area di configurazione della bolla e dello specifico pulsante tramite l'utilizzo della classe `Monolith::UI::Bubbles::BubbleDiscriminator`.

Metodi:

- `+doMakeBubbleSender() : MeteoBubble`
Metodo che gestisce la creazione della bolla vista dal mittente.
- `+doMakeBubbleReceiver() : MeteoBubble`
Metodo che gestisce la creazione della bolla vista dal ricevente.
- `+doMakeConfigurationMenu() : MeteoBubbleConfig`
Metodo che gestisce la creazione dell'area di configurazione della bolla.
- `+doMakeButton() : MeteoBubbleCreationButton`
Metodo che gestisce la creazione dello specifico pulsante da inserire nel menu iniziale di creazione. Viene lasciata l'implementazione della super classe.

Applicazioni

Viene utilizzata per gestire la creazione della specifica bolla meteo, della specifica area di configurazione della bolla e dello specifico pulsante.



3.5.25 MeteoBubbleReceiver

ComponenteMeteoBubble

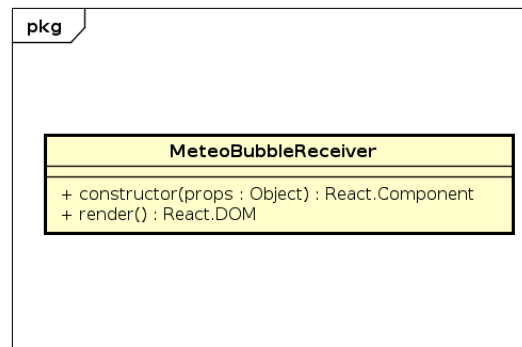


Figura 82: Diagramma per MeteoBubbleReceiver in

Descrizione

Istanziatura concreta della classe Monolith::Bubbles::Bubble che rappresenta la bolla vista dal ricevente.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM** Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata per la visualizzazione della bolla da parte del ricevente. La classe è composta da Monolith::UI::UI-Layouts::HorizontalLayout (composta a sua volta da MeteoItem) che dispone i componenti visuali uno accanto all'altro.



3.5.26 MeteoBubbleConfigMenu

ComponenteMeteoBubble

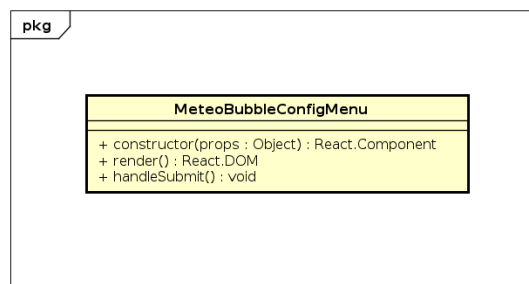


Figura 83: Diagramma per MeteoBubbleConfigMenu in

Descrizione

Istanziatura concreta della classe `Monolith::UI::Bubbles::BubbleConfig` che rappresenta l'area di configurazione della bolla.

Metodi:

- `+constructor(props : Object) : React.Component`
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- `+render() : React.DOM` Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento `React` che può essere una rappresentazione di un componente `DOM` nativo o un altro componente composito.
- `+handleSubmit() : void` Metodo che viene invocato quando la configurazione della bolla è terminata ed è pronta per essere inviata.

Applicazioni

Viene utilizzata per la visualizzazione dell'area di configurazione della bolla. La classe è composta da `Monolith::UI::UI-SingleComponents::LineEditPushButton`.



3.5.27 ResultsViewer

ComponenteSurveyBubble

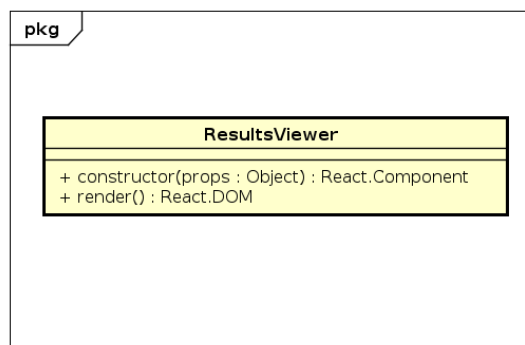


Figura 84: Diagramma per ResultsViewer in

Descrizione

Componente React che rappresenta il risultato del sondaggio.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM** Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzato per la visualizzazione del risultato del sondaggio.



3.5.28 SurveyManager

ComponenteSurveyBubble

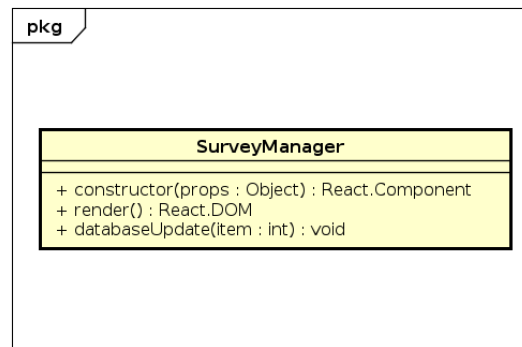


Figura 85: Diagramma per SurveyManager in

Descrizione

Classe predisposta per la gestione del sondaggio.

Metodi:

- `+constructor(props : Object) : React.Component`
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- `+render() : React.DOM` Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.
- `+databaseUpdate(item : int) : void` Metodo che gestisce l'aggiornamento del database al cambiare dei voti nel sondaggio.

Applicazioni

Viene utilizzata per gestire l'aggiornamento del database al cambiare dei voti nel sondaggio. È composta da `Monolith::UI::UI-Layouts::VerticalLayout` (a sua volta composta da `Monolith::UI::UI-SingleComponents::Label` e `Monolith::UI::UI-SingleComponents::RadioButtonGroup`, `ResultsViewer`) che dispone i componenti visuali uno sotto l'altro.



3.5.29 SurveyBubbleSender

ComponenteSurveyBubble

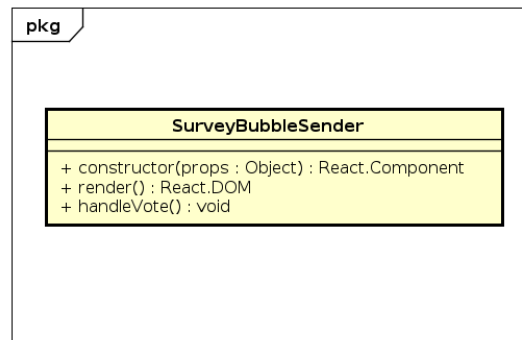


Figura 86: Diagramma per SurveyBubbleSender in

Descrizione

Istanziatura concreta della classe `Monolith::Bubbles::Bubble` che rappresenta la bolla vista dal mittente.

Metodi:

- `+constructor(props : Object) : React.Component`
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- `+render() : React.DOM` Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento `React` che può essere una rappresentazione di un componente `DOM` nativo o un altro componente composito.
- `+handleVote() : void` Metodo che viene invocato quando è stata selezionata un'opzione.

Applicazioni

Viene utilizzata per la visualizzazione della bolla da parte del mittente.



3.5.30 SurveyBubbleCreator

ComponenteSurveyBubble

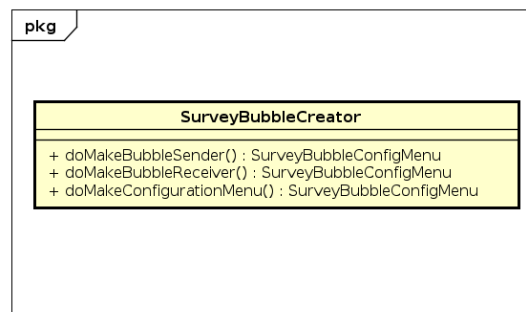


Figura 87: Diagramma per SurveyBubbleCreator in

Descrizione

Istanziatura concreta della classe Monolith::UI::Bubbles::BubbleCreator che gestisce la creazione della specifica bolla sondaggio, della specifica area di configurazione della bolla e dello specifico pulsante tramite l'utilizzo della classe Monolith::UI::Bubbles::BubbleDiscriminator.

Metodi:

- +doMakeBubbleSender() : SurveyBubble
Metodo che gestisce la creazione della bolla vista dal mittente.
- +doMakeBubbleReceiver() : SurveyBubble
Metodo che gestisce la creazione della bolla vista dal ricevente.
- +doMakeConfigurationMenu() : SurveyBubbleConfig
Metodo che gestisce la creazione dell'area di configurazione della bolla.
- +doMakeButton() : SurveyBubbleCreationButton
Metodo che gestisce la creazione dello specifico pulsante da inserire nel menu iniziale di creazione. Viene lasciata l'implementazione della super classe.

Applicazioni

Viene utilizzata per gestire la creazione della specifica bolla sondaggio, della specifica area di configurazione della bolla e dello specifico pulsante.



3.5.31 SurveyBubbleReceiver

ComponenteSurveyBubble

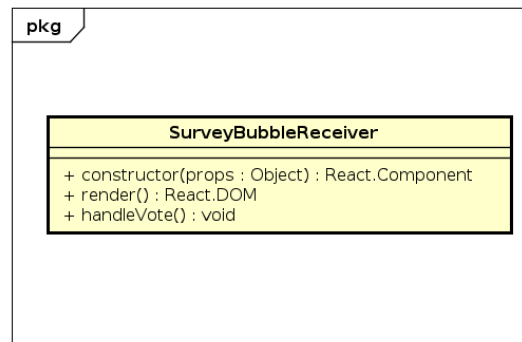


Figura 88: Diagramma per SurveyBubbleReceiver in

Descrizione

Istanziatura concreta della classe `Monolith::Bubbles::Bubble` che rappresenta la bolla vista dal ricevente.

Metodi:

- `+constructor(props : Object) : React.Component`
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- `+render() : React.DOM` Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento `React` che può essere una rappresentazione di un componente `DOM` nativo o un altro componente composito.
- `+handleVote() : void` Metodo che viene invocato quando è stata selezionata un'opzione.

Applicazioni

Viene utilizzata per la visualizzazione della bolla da parte del ricevente.



3.5.32 SurveyBubbleConfigMenu

ComponenteSurveyBubble

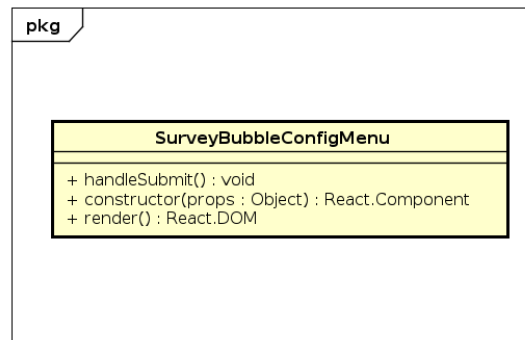


Figura 89: Diagramma per SurveyBubbleConfigMenu in

Descrizione

Istanziatura concreta della classe `Monolith::UI::Bubbles::BubbleConfig` che rappresenta l'area di configurazione della bolla.

Metodi:

- `+constructor(props : Object) : React.Component`
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- `+render() : React.DOM` Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento `React` che può essere una rappresentazione di un componente `DOM` nativo o un altro componente composito.
- `+handleSubmit() : void` Metodo che viene invocato quando la configurazione della bolla è terminata ed è pronta per essere inviata.

Applicazioni

Viene utilizzata per la visualizzazione dell'area di configurazione della bolla. La classe è composta da `Monolith::UI::UI-Layouts::VerticalLayout` (composta a sua volta da `Monolith::UI::UI-SingleComponents::LineEdit`, `Monolith::UI::UI-SingleComponents::PushButton`, `Monolith::UI::UI-SingleComponents::LabelEditPushButton`) che dispone i componenti visivi uno sotto l'altro.



3.5.33 MessageTranslation

ComponenteTranslationBubble

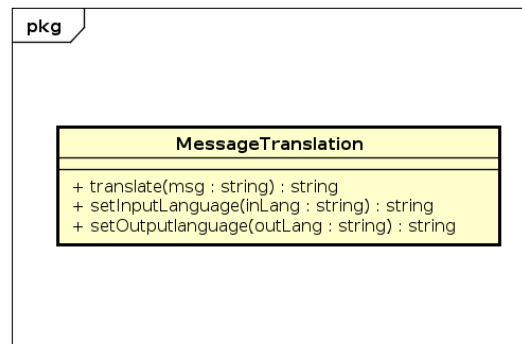


Figura 90: Diagramma per MessageTranslation in

Descrizione

Classe predisposta per l'effettiva traduzione del messaggio.

Metodi:

- **+translate(msg : string) : string**
Metodo che effettua la traduzione del messaggio utilizzando l'utility Polygloth.js.
- **+setInputLanguage(inLang : string) : void**
Metodo che imposta la lingua di partenza.
- **+setOutputLanguage(outLang : string) : void**
Metodo che imposta la lingua di arrivo.

Applicazioni

Viene utilizzata per effettuare la traduzione del messaggio di testo dalla lingua di partenza a quella di arrivo. Si serve dell'utility Polygloth.js.



3.5.34 TranslationBubbleSender

ComponenteTranslationBubble

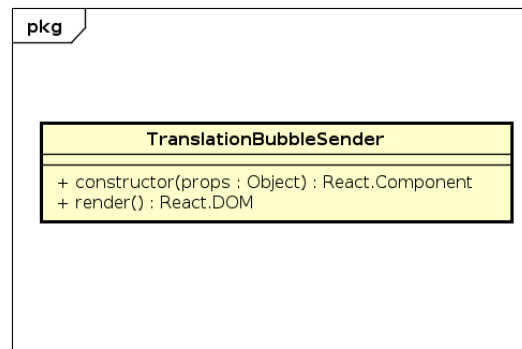


Figura 91: Diagramma per TranslationBubbleSender in

Descrizione

Istanziatura concreta della classe Monolith::Bubbles::Bubble che rappresenta la bolla vista dal mittente.

Metodi:

- +constructor(props : Object) : React.Component
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- +render() : React.DOM Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzato per la visualizzazione della bolla da parte del mittente. La classe è composta da Monolith::UI::UI-Layouts::VerticalLayout (composta a sua volta da Monolith::UI::UI-SingleComponents::Label) che dispone i componenti visuali uno sotto l'altro.



3.5.35 TranslationBubbleCreator

ComponenteTranslationBubble

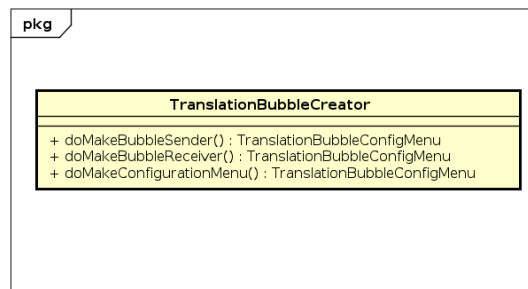


Figura 92: Diagramma per TranslationBubbleCreator in

Descrizione

Istanziatura concreta della classe Monolith::UI::Bubbles::BubbleCreator che gestisce la creazione della specifica bolla traduttore, della specifica area di configurazione della bolla e dello specifico pulsante tramite l'utilizzo della classe Monolith::UI::Bubbles::BubbleDiscriminator.

Metodi:

- +doMakeBubbleSender() : TranslationBubble
Metodo che gestisce la creazione della bolla vista dal mittente.
- +doMakeBubbleReceiver() : TranslationBubble
Metodo che gestisce la creazione della bolla vista dal ricevente.
- +doMakeConfigurationMenu() : TranslationBubbleConfig
Metodo che gestisce la creazione dell'area di configurazione della bolla.
- +doMakeButton() : TranslationBubbleCreationButton
Metodo che gestisce la creazione dello specifico pulsante da inserire nel menu iniziale di creazione. Viene lasciata l'implementazione della super classe.

Applicazioni

Viene utilizzata per gestire la creazione della specifica bolla traduttore, della specifica area di configurazione della bolla e dello specifico pulsante.



3.5.36 TranslationBubbleReceiver

ComponenteTranslationBubble

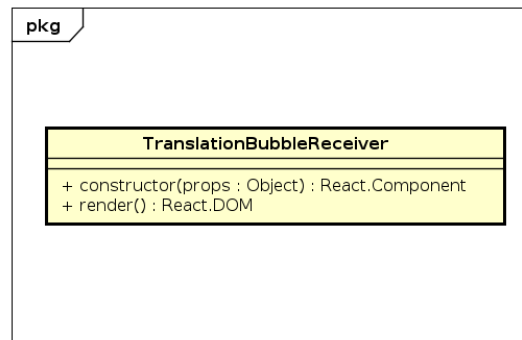


Figura 93: Diagramma per TranslationBubbleReceiver in

Descrizione

Istanziatura concreta della classe Monolith::Bubbles::Bubble che rappresenta la bolla vista dal ricevente.

Metodi:

- **+constructor(props : Object) : React.Component**
Costruttore della sottoclasse di React.Component in cui è necessario chiamare super (props) ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- **+render() : React.DOM** Metodo che esamina this.props e this.state e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.

Applicazioni

Viene utilizzata per la visualizzazione della bolla da parte del ricevente. La classe è composta da Monolith::UI::UI-Layouts::VerticalLayout (composta a sua volta da Monolith::UI::UI-SingleComponents::Label) che dispone i componenti visuali uno sotto l'altro.



3.5.37 TranslationBubbleConfigMenu

ComponenteTranslationBubble

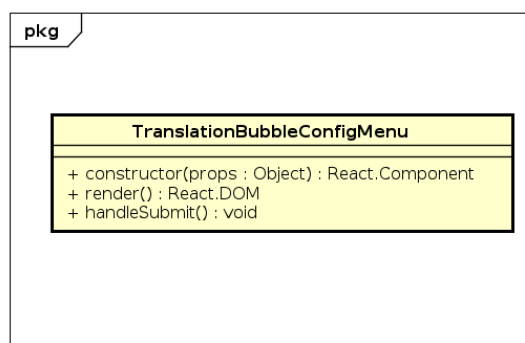


Figura 94: Diagramma per TranslationBubbleConfigMenu in

Descrizione

Istanziatura concreta della classe `Monolith::UI::Bubbles::BubbleConfig` che rappresenta l'area di configurazione della bolla.

Metodi:

- `+constructor(props : Object) : React.Component`
Costruttore della sottoclasse di `React.Component` in cui è necessario chiamare `super (props)` ed è possibile inizializzare lo stato per i dati soggetti a cambiamento.
- `+render() : React.DOM` Metodo che esamina `this.props` e `this.state` e restituisce un singolo elemento React che può essere una rappresentazione di un componente DOM nativo o un altro componente composito.
- `+handleSubmit() : void` Metodo che viene invocato quando la configurazione della bolla è terminata ed è pronta per essere inviata.

Applicazioni

Viene utilizzata per la visualizzazione dell'area di configurazione della bolla. La classe è composta da `Monolith::UI::UI-Layouts::VerticalLayout` (composta a sua volta da `Monolith::UI::UI-SingleComponents::TextAreaComboBox`, `Monolith::UI::UI-SingleComponents::ComboBox` e `Monolith::UI::UI-SingleComponents::PushButton`) che dispone i componenti visivi uno sotto l'altro.

4 Standard di Progetto

4.1 Standard di documentazione del codice

Gli standard di documentazione del codice sono definiti nel documento `NormeDiProgetto`



4.2 Standard di denominazione di entità e relazioni

Gli standard di denominazione sono definiti nel documento NormeDiProgetto

4.3 Strumenti di lavoro

Gli strumenti da utilizzare e le procedure da seguire sono descritti nel documento NormeDiProgetto

5 Diagrammi di Attività

Il diagramma delle attività è un diagramma definito all'interno dello Unified Modeling Language (UML) che definisce le attività da svolgere per realizzare una data funzionalità. Può essere utilizzato durante la progettazione del software per dettagliare un determinato algoritmo. Più in dettaglio, un activity diagram definisce una serie di attività o flusso, anche in termini di relazioni tra le attività, i responsabili per le singole attività e i punti di decisione. L'activity diagram è spesso usato come modello complementare allo Use Case Diagram, per descrivere le dinamiche con cui si sviluppano i diversi use case.

5.1 Configurazione sondaggio

Questo diagramma rappresenta la configurazione e l'invio della bolla sondaggio. L'utente inserisce il titolo del sondaggio, dopodiché avviene l'inserimento delle opzioni. L'utente ha la possibilità di inserire più opzioni (rappresentato nel diagramma con la freccia che parte dal nodo decisione e raggiunge il nodo azione chiamato inserisci opzioni). Una volta compiuto il processo di configurazione l'utente decide di inviare la bolla (rappresentato dal nodo invia). Se il numero delle opzioni inserite è minore di due allora il flusso ritorna al nodo azione chiamato "inserisci opzione" per poter permettere all'utente di incrementare il numero delle opzioni del sondaggio. Altrimenti (il numero di opzioni è maggiore di uno) il flusso arriva al nodo di fine dell'attività.

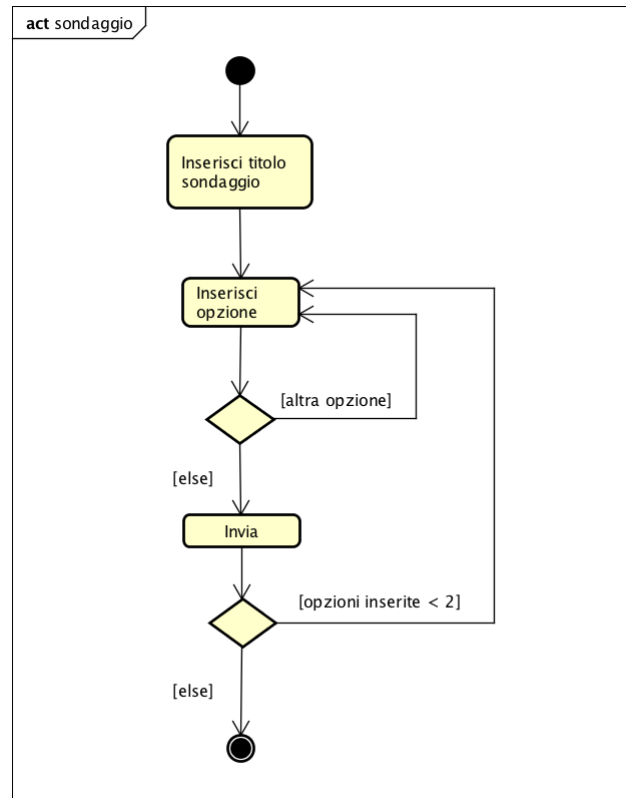


Figura 95: Diagramma di attività per la bolla sondaggio

5.2 Configurazione ListBubble

Questo diagramma rappresenta la configurazione della bolla ListBubble. Il flusso inizia con l'inserimento del titolo della lista. Dopo questo processo il flusso raggiunge il nodo decisione dove l'utente sceglie il modo di inserimento degli elementi. L'inserimento degli elementi avviene o manualmente (inserisci elemento manualmente) oppure con la selezione nella lista degli elementi predefiniti (inserisci elemento da checklist). Nel caso l'utente scegliesse di inserire un elemento dalla checklist predefinita, il flusso raggiunge il nodo decisione. Da questo punto l'utente può scegliere di inviare la bolla già configurata e il flusso raggiunge il nodo terminale, oppure scegliere di ritornare nel nodo decisione raggiunto precedentemente per poter inserire un altro elemento. Anche nel caso in cui l'utente scegliesse di inserire un elemento manualmente il flusso segue lo stesso percorso, ovvero l'utente inserisce l'elemento e decide di inviare la bolla con il raggiungendo il nodo terminale oppure inserire un altro elemento.

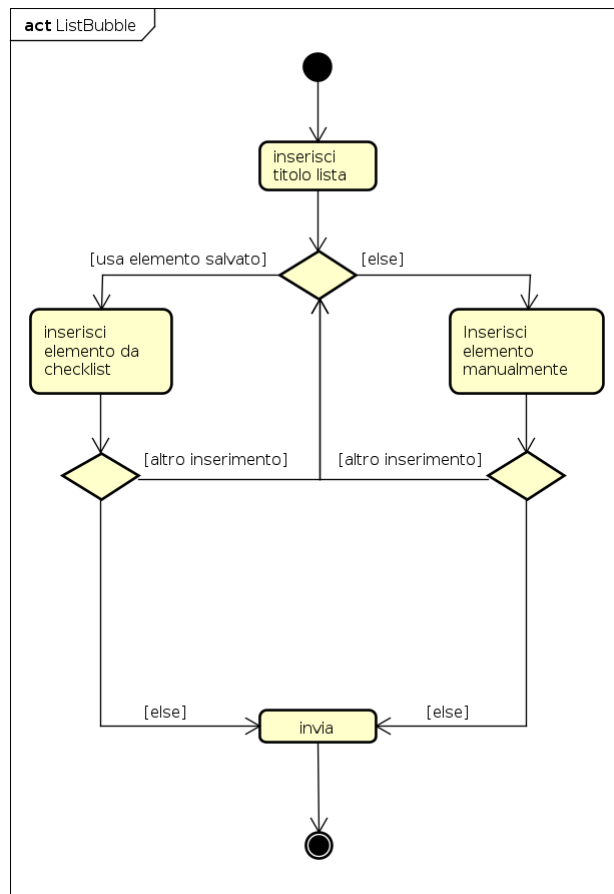


Figura 96: Diagramma di attività per la bolla ListBubble

6 Diagramma di Sequenza

6.1 Creazione Bolla

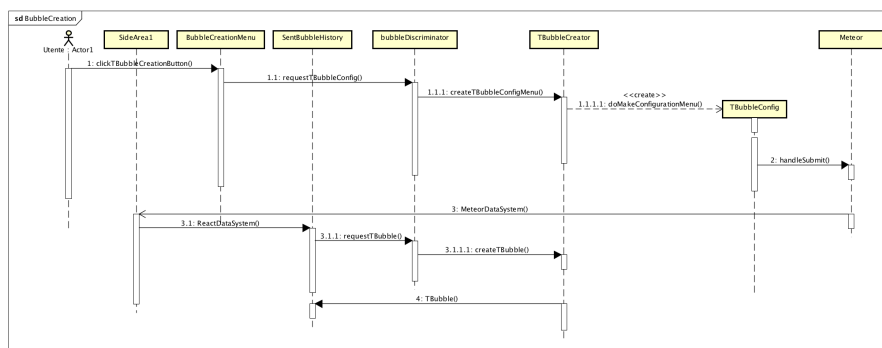


Figura 97: Diagramma di sequenza per la creazione di una bolla



Scenario: Questo scenario rappresenta la creazione di una bolla con l'utente invia la richiesta per accedere al menù di creazione. La richiesta passa per il bubbleDiscriminator che capisce la tipologia di bolla e passa la richiesta al TBubbleCreator e TBubbleConfig che costruiscono il menù di configurazione adatto. Una volta compilati i campi la richiesta torna al bubbleDiscriminator e al TBubbleCreator che creano la bolla aggiungendola alla SentBubbleHistory.

6.2 Cancellazione Bolla

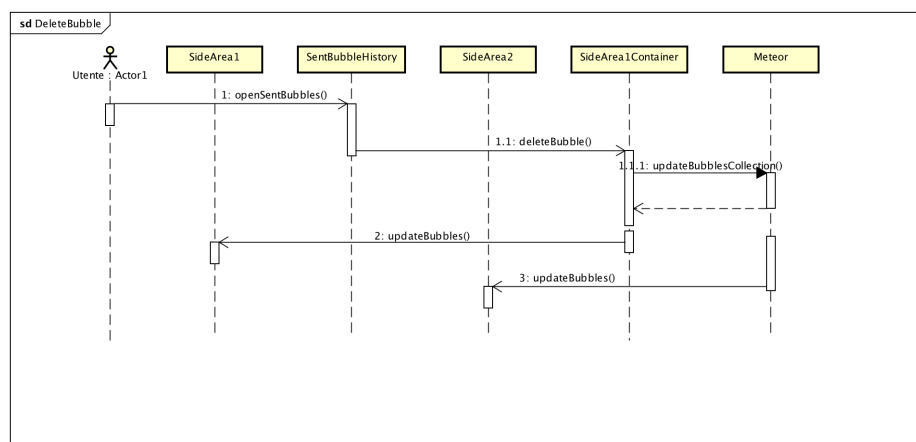


Figura 98: Diagramma di sequenza per la cancellazione di una bolla

Scenario: Questo scenario rappresenta la cancellazione, da parte di un utente, di una delle proprie bolle inviate precedentemente. L'utente in questione accede alla propria area di bolle inviate e seleziona la bolla da cancellare. La richiesta arriva al database dove avviene la cancellazione, successivamente, Meteor, gestisce l'aggiornamento dei dati nello storico delle bolle inviate (dell'utente attore) e in quello delle bolle ricevute degli altri utenti.



6.3 Voto Bolla-Sondaggio

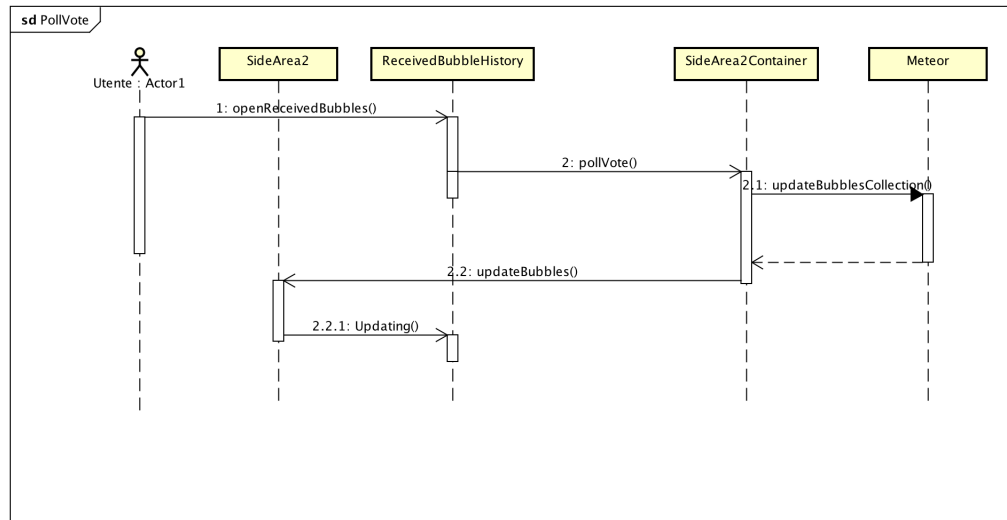


Figura 99: Diagramma di sequenza per il voto in una bolla-sondaggio

Scenario: Questo scenario rappresenta il voto di un utente ad una bolla sondaggio. L'utente accede all'area delle bolle ricevute e vota l'opzione che desidera, il voto viene salvato nel database e, attraverso Meteor, vengono aggiornati i dati della bolla in questione di ogni utente.

7 Tracciamento

7.1 Tracciamento componenti-requisiti

Componente	Requisiti
Monolith::UI::UI-Layouts	11.2.1.2.1
	11.2.1.2.2
	11.2.1.2.3
Monolith::UI::UI-SingleComponents	11.2.1
	11.2.1.1
	11.2.1.1.1
	11.2.1.1.3
	11.2.1.1.4
	11.2.1.1.5
	11.2.1.1.6
	11.2.1.2

Tabella 3: Tracciamento componenti - requisiti



7.2 Tracciamento requisiti-componenti

Requisito	Componente
11.2.1	Monolith::UI::UISingleComponents
11.2.1.1	Monolith::UI::UISingleComponents
11.2.1.1.1	Monolith::UI::UISingleComponents
11.2.1.1.3	Monolith::UI::UISingleComponents
11.2.1.1.4	Monolith::UI::UISingleComponents
11.2.1.1.5	Monolith::UI::UISingleComponents
11.2.1.1.6	Monolith::UI::UISingleComponents
11.2.1.2	Monolith::UI::UISingleComponents
11.2.1.2.1	Monolith::UI::UILayouts
11.2.1.2.2	Monolith::UI::UILayouts
11.2.1.2.3	Monolith::UI::UILayouts

Tabella 5: Tracciamento requisiti - componenti

7.3 Tracciamento classi-requisiti

Class	Requisiti
-------	-----------

Tabella 7: Tracciamento classi - requisiti

7.4 Tracciamento requisiti-classi

Requisiti	Classi
-----------	--------

Tabella 9: Tracciamento requisiti - classi

A Descrizione Design Pattern

Un design pattern è una soluzione progettuale elegante e generale ad un problema ricorrente. In particolare si tratta di un modello logico da applicare per la risoluzione di un problema che può presentarsi in diverse situazioni durante la fase di progettazione e sviluppo del software, ancora prima della definizione dell'algoritmo risolutivo.

Essi si suddividono in quattro categorie:

- **Architetturali** : esprimono schemi di base per impostare l'organizzazione strutturale di un sistema software;



- **Creazionali** : forniscono un'astrazione del processo di istanziazione degli oggetti;
- **Strutturali** : si occupano delle modalità di composizione di classi e oggetti per formare strutture complesse;
- **Comportamentali** : si occupano di algoritmi e dell'assegnamento di responsabilità tra oggetti collaboranti.

A.1 Design Pattern Utilizzati

A.1.1 Factory Method

Rappresenta uno dei pattern creazionali adottati dal gruppo Obelix, esso indirizza il problema della creazione di oggetti senza specificarne l'esatta classe. Questo pattern raggiunge il suo scopo fornendo un'interfaccia per creare un oggetto, ma lascia che le sottoclassi decidano quale oggetto istanziare.

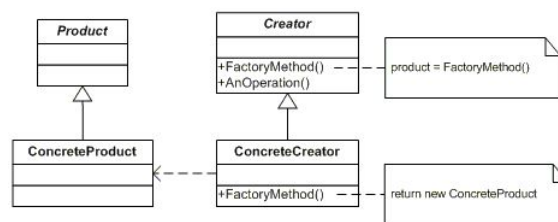


Figura 100: Diagramma del Factory method

Product definisce l'interfaccia implementata da ConcreteProduct, creator definisce il factory method che restituisce una interfaccia di tipo product, ConcreteCreator definisce il metodo factory effettivo per la creazione di un'istanza particolare di tipo Product.

I motivi che portano alla scelta del suo utilizzo sono:

- La creazione di un oggetto preclude il suo riuso senza una significativa duplicazione di codice
- La creazione di un oggetto richiede l'accesso ad informazioni o risorse che non dovrebbero essere contenute nella classe di composizione
- La gestione del ciclo di vita degli oggetti gestiti deve essere centralizzata in modo da assicurare un comportamento coerente all'interno dell'applicazione