4OURSQUARED

4oursquared.unipd@gmail.com

Lumos Minima

Imola Informatica

Norme di Progetto

Informazioni	
Redattori	Soldà Matteo
Versione	0.0.2
Uso	interno

Descrizione

Questo documento contiene le procedure, gli strumenti e i criteri di qualità che verranno usati nel progetto.

Versione	Data	Redattore	Verificatore	Descrizione
0.0.2	20/04/2023	Ceccato Francesco	Soldà Matteo	Terminata Sezione Convenzioni di Codifica
0.0.1	19/04/2023	Soldà Matteo		Terminata Sezione Documentazione
0.0.0	24/03/2023	Soldà Matteo	Brotto Romina	Prima stesura.

Contents

1 Introduzione

1.1 Scopo del Documento

Il documento si prefigge lo scopo di definire i metodi e le attività chiave legate al *Way of Working*. Tutti i membri del gruppo si impegnano a seguire quanto riportato.

2 Processi Primari

2.1 Fornitura

2.1.1 Repository Pubblica

Durante lo sviluppo del progetto, il codice e la documentazione saranno depositati in una repository pubblica. La parte di interesse del proponente e del committente riguarda la cartella *public*.
La cartella sopra citata contiene la documentazione necessaria alle revisioni di periodo con il committente. Tale cartella rispecchia la struttura della cartella *docs*, situata nella parte privata.
I file che prevedono versionamento, riporteranno al loro interno la numerazione relativa alla versione attuale e un riassunto delle versioni precedenti.
Ogni modifica della parte pubblica deve essere approvata, tramite *pull request*, dal responsabile.

2.1.2 Contatti

Tutti i contatti ufficiali, sia con il proponente che con il committente, avverranno tramite la mail del gruppo 4oursquared.unipd@gmail.com.
Le mail per la presentazione ad una revisione di periodo necessita di una conferma unanime, valutata la preparazione. Questa mail deve contenere la lettera di presentazione con i link ai vari documenti aggiunti/aggiornati nella parte pubblica della repository.

2.2 Sviluppo

2.2.1 Organizzazione dei File

Tutta la documentazione sarà contenuta all'interno della cartella *docs/*. Questa sarà inoltre suddivisa in *interna* ed *esterna*, le quali conterranno rispettivamente la documentazione interna al gruppo e quella da condividere con il proponente e con il committente.
I verbali interni saranno conservati nella sottocartella *docs/interni/verbali* mentre quelli esterni saranno conservati in *docs/esterni/verbali*.
Per ogni documento esisterà una nuova sottocartella omonima.

2.2.2 Convezioni di Codifica

Convenzioni Linguistiche

- Ogni elemento del codice deve essere scritto in lingua inglese, ad eccezione dei commenti, che dovrebbero essere scritti in italiano per evitare incomprensioni.

Convenzioni di Nomenclatura File

- Un file che contiene l'interfaccia o l'implementazione di una sola ed esclusiva classe (ad eccezione di classi interne e sottoclassi) dovrebbe essere individuato nel seguento modo: `NomeClasse.estensione`.

Convenzioni Stilistiche

	Java	JavaScript	Python	TypeScript
Funzioni e Metodi	camelCase()	camelCase()	snake_case()	camelCase()
Classi	PascalCase			
Interfacce	PascalCase	PascalCase	NA	PascalCase
Namespace e Package	lowercase	NA	NA	camelCase
Costanti	SCREAMING_SNAKE_CASE			
Variabili Locali e Attributi	camelCase	camelCase	snake_case	snake_case
Variabili Globali	NA	camelCase	snake_case	snake_case

A seguire, in JavaScript, le funzioni e i metodi privati anteporranno il simbolo di underscore prima del nome.
Per quanto riguarda invece i linguaggi di markup, i tag e gli attributi andranno scritti in lowercase se primitivi o in kebab-case se custom o complessi.
Inoltre:

- I membri di una classe dovrebbero essere dichiarati e/o definiti nel seguente ordine:
  - Variabili di classe (statiche)
  - Variabili di istanza (attributi)
    Nel seguente ordine (ove possibile):
    1. public
    2. protected
    3. private
  - Costruttore
  - Distruttore (se richiesto)
  - Metodi
- Un blocco di codice delimitato da parentesi graffe dovrebbe essere strutturato apponendo le parentesi graffe ognuna in una nuova riga;
- Si predilige l'uso della tabulazione per indentare la porzione di codice inclusa in un blocco.

CSS

- Mantenere un file chiamato `nomepagina.css` per la versione desktop di ciascuna pagina;
- Mantenere un file chiamato `global.css` per le regole che si applicano ad ogni aspetto del sito, in particolare ai moduli universali quali, a titolo esemplificativo e non esaustivo, header e footer;
- Mantenere un file chiamato `mini.css` con accorgimenti per la versione mobile
- Specificare le unità di misura in *em*, *rem*, *%*, *vw*, *vh* (unità relative), a parte per i valori delle seguenti proprietà, i quali possono essere espressi in *px*:
  - *text-shadow*
  - *box-shadow*

3 Processi Secondari

3.1 Documentazione

3.1.1 Ciclo di Vita

Ogni documento seguirà il seguente flusso:

- **Pianificazione**
  Il documento viene progettato ad alto livello secondo quanto richiesto.
- **Redazione**
  Un membro del gruppo, nel ruolo di redattore, stila il documento.
- **Revisione**
  Un membro del gruppo, nel ruolo di revisore, controlla l'assenza di errori grammaticali e il contenuto sia conforme alle norme di progetto.
- **Approvazione**
  Un membro del gruppo, nel ruolo di responsabile, verifica che il contenuto del documento sia corretto.

3.1.2 Strumenti Utilizzati

Per la stesura dei documenti verrà utilizzato il linguaggio LaTeX.
Affinché i documenti risultino stilisticamente coerenti, dovranno includere al loro interno il file *docs/template/style.tex* per lo stile generale e il file *docs/template/front_page.tex* per utilizzare il template della prima pagina.

3.1.3 Struttura

Ogni documento sarà così strutturato:

1. Pagina di intestazione
2. Pagina con lista delle versioni, ove necessario
3. Pagina indice, ove necessario
4. Una o più pagine di contenuto

Le varie sezioni saranno così definite:

Pagina di Intestazione

- Logo e nome del gruppo
- Nome del progetto e azienda cliente
- Mail di contatto
- Titolo del documento
- Info generali
  - Redattori
  - Revisori
  - Responsabile
  - Versione, ove necessario
  - Destinazione d'uso (interno o esterno)
- Sinossi

Elenco delle Versioni
L'elenco delle versioni è una tabella così definita:

- Versione
- Data
- Redattore
- Revisore
- Descrizione

Le righe della tabella sono organizzate in ordine cronologico inverso.

Indice
L'indice è stato creato utilizzando il comando $\backslash tableofcontents$

Norme di Progetto

- **Scopo:** Lo scopo del documento è quello di definire le procedure, gli strumenti e criteri di qualità al fine di stabilire un *Way of Working*. Ogni membro del gruppo sarà tenuto a rispettare le indicazioni del documento presentate
- **Titolo:** Norme di Progetto
- **Nome del File:** *norme_di_progetto.tex*

Verbali

- **Scopo:** Lo scopo dei verbali è quello di rendicontare ciò che viene discusso durante una riunione, sia interna che esterna, delineando gli impegni che ne derivano
- **Titolo:** Ogni verbale sarà titolato con "$Verbale\ del\ <data>\ [con <esterni>]$"
- **Nome del File:** Tutti i file saranno chiamati $<yyyy>\_<mm>\_<dd>\_<tipo>.tex$ dove $<yyyy>\_<mm>\_<dd>$ indica la data nella quale si è effettuatal a riunione nel formato anno-mese-giorno; ¡tipo¿ indica se la riunione è avvenuta tra i soli membri del gruppo "I" o con persone esterne "E".
- **Inidice:** Non presente in quanto non utile
- **Struttura:**
  - Data, ora e durata;
  - Luogo;

  - Partecipanti (interni ed esterni, raggruppati per appartenenza e ordinati alfabeticamente secondo il cognome);
  - Ordine del giorno;
  - Conclusioni derivanti dalla riunione;
  - Impegni assunti.
- **Stesura:** All'inizio di ogni riunione verrà nominato un membro del gruppo che redigerà il verbale e un membro che lo validerà. Chi redige il verbale avrà 24 ore per completare il lavoro, lo stesso tempo sarà concesso al validatore per approvarlo e segnalare le modifiche da applicare.

Piano di Progetto

- **Scopo:** Lo scopo del documento è quello di definire gli obiettivi da raggiungere, stabilendo le tempistiche e il responsabile, tenendo traccia dei progeressi e valutando i coste sostenuti rispetto a quelli preventivati.
- **Titolo:** Piano di Progetto
- **Nome del File:**

Piano di Qualifica

- **Scopo:** Lo scopo del documento è quello di definire le metriche e i requisiti minimi di qualità affinchè un prodotto del progetto possa essere approvato.
- **Titolo:** Piano di Qualifica
- **Nome del File:**

3.1.4 Convenzioni Documentali

- **Riferimento a file o cartelle:**
  - Per fare riferimento al nome di un file o di una cartella si utilizza il `testo monospaziato`;
  - Il nome di una cartella deve sempre terminare con `/`.
- **Stringhe e nomi:**
  - Le stringhe vanno scritte tra doppi apici;
  - Per indicare un parametro rappresentate una stringa si usa il testo tra parentesi angolari. Tale parametro non può contenere spazi;
  - Per indicare una parte di nome o stringa opzionale si usa il testo racchiuso tra parentesi quadre.
- **Riferimento tra documenti:**
  - Riferimenti interni al documento: comando `\ref`;
  - Riferimenti esterni al documento: nome della sezione del documento a cui si fa riferimento in corsivo.

3.2 Versionamento

3.2.1 Repository Github

Per il versionamento del progetto si è deciso di utilizzare *Git* sulla piattaforma *GitHub*. La repository è raggiungibile tramite l'indirizzo https://github.com/4ourSquared/LumosMinima.

3.2.2 Convenzioni di Versionamento

Per il versionamento si farà uso del Versionamento Semantico, secondo le linee guida. Questo tipo di versionamento si fonda sulle seguenti caratteristiche principali:

1. Un numero di versione deve essere nella forma $X.Y.Z$ dove X, Y, Z sono numeri interi non negativi e non devono contenere zeri iniziali;
2. La versione *major zero* (es: $0.X.Y$) serve solo per lo sviluppo iniziale e indica una versione non stabile del prodotto;
3. La versione *patch Z* (es: $x.y.Z$) deve essere incrementata solo se sono state introdotte correzioni retrocompatibili del testo (modifica da parte del reddatore);
4. La versione *minor Y* (es: $x.Y.z$) deve essere incrementata se nella documentazione è stata introdotta una nuova modifica retrocompatibile (approvazione da parte del verificatore);
5. la versione *major X* (es: $X.y.z$) deve essere incrementata solo se nella documentazione è stata introdotta una modifica non retrocompatibile con le versioni precedenti (pubblicazione da parte del responsabile).

3.2.3 Comandi Utili

- **Sincronizzazione con la repository remota:** `git pull`;
- **Creazione di un nuovo branch locale:** `git branch <nome_branch>`;
- **Passaggio ad un branch locale esistente:** `git checkout <nome_branch>`;
- **Aggiunta delle modifiche alla stage area:** `git add .`;
- **Creazione del commit con le modifiche:** `git commit -m <descrizione>`;
- **Push sul remote in un branch remoto non esistente:** `git push --set-upstream origin <nome_branch>`;
- **Push sul branch remoto esistente omonimo del branch locale:** `git push`.

Escludendo per il momento l'utilizzo di *git flow*, questi risultano i comandi da utilizzare.