Tecnologie Web (6 CFU)

C.d.L. in Informatica

Compito del 27 gennaio 2012

Nome:

Cognome:

Matricola:

Team:

□ Non è la prima volta che provo questo esame

Ricapitolo: fare qui sotto una croce sul numero di ciascun esercizio risolto. Se lo si consegna su un foglio protocollo, scrivere in maniera chiara il numero dell'esercizio presso la sua soluzione.

Esercizio		**Punti**	**Voto**
1	Domande di base	12	
2	HTML	4	
3	Javascript	8	
4	Semantic Web	6	
5	Teoria	4	
Totale		**34**	

Domande di base (12 punti)

Rispondere correttamente ad almeno tre delle seguenti domande:

A. La frase "CSS è un linguaggio di markup" è vera o falsa? Motivare la risposta.

B. Quale fra i seguenti elementi HTML non è un componente di un form:

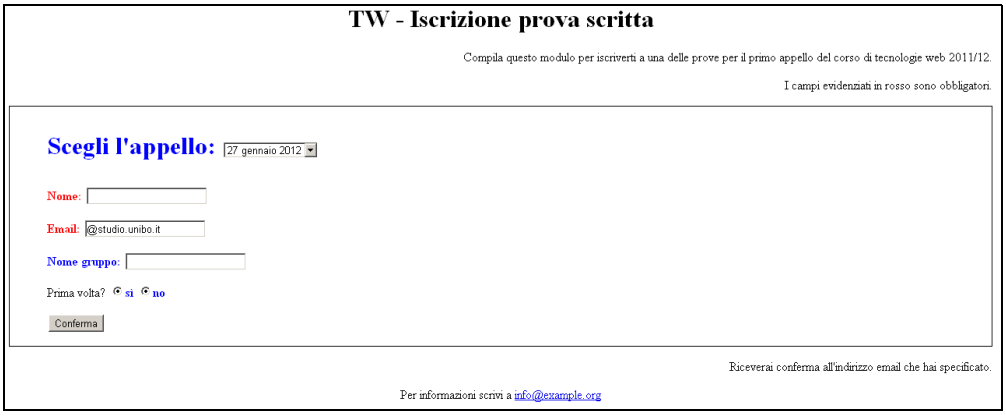
- input
- label
- button
- title
- fieldset

C. Scrivere un semplice template XSLT che trasforma gli elementi "p" in elementi "paragrafo" e ne preserva il contenuto.

D. Fornire un esempio di array associativo in Javascript.

HTML (4 punti)

Scrivere il codice HTML 5 (e CSS) di un documento che rappresenti il contenuto di questa immagine:



Nello scrivere il codice si tengano in considerazione questi vincoli:

- l'URI al quali il modulo dovrà inviare i dati in POST è `http://example.org/tw/iscrizioni`;
- chiede all'utente i seguenti campi (etichetta, caratteristiche etichetta, nome campo, tipo di dato):
    - "Scegli l'appello:", grassetto blu di dimensione doppia, "data", scelta al massimo di una delle seguenti possibilità:
        - "27 gennaio 2012", "20120127"
        - "18 febbraio 2012", "20120218"
    - "Nome: ", grassetto rosso, "nome", testo libero
    - "Email: ", grassetto rosso, "email", testo libero
    - "Nome gruppo: ", grassetto blu, "gruppo", testo libero
    - "Conferma", conferma, pulsante d'invio modulo
- ed inoltre il campo "Prima volta?" permette di scegliere fra una di queste possibilità (etichetta, caratteristiche etichetta, nome campo):
    - "sì", blu, "prima"
    - "no", blu, "seconda"

Tutti i dettagli non specificati (per esempio nomi file e variabili) possono essere liberamente scelti.

Inoltre,

- non è possibile fare uso di tabelle;
- l'elemento presentazionale `b` non può essere usato.
- nessun elemento deve contenere l'attributo `class` (di conseguenza non si possono usare usare selettori classe nel codice CSS);
- inoltre non si devono utilizzare selettori di id nel codice CSS;
- nessun elemento deve contenere l'attributo `style`;

Si consiglia di scrivere tutto il codice CSS in un "file" separato, non in elementi `<style>`

Javascript (8 punti)

Un forum online vuole attivare un meccanismo online per l'aggiornamento automatico dei messaggi delle discussioni.

La discussione è identificata da un id della forma "thread_12345" posto nella variabile globale thread_id. I messaggi di una discussione sono tutti visualizzati in lista all'interno di un div con id "MessageList", ciascuno inserito in un div con classe "message" e id unico (ad esempio "message_12345"), all'interno del quale c'è un p di classe "user" con il nome dell'autore (cliccabile, porta all'URL "http://www.forum.com/user/[user_id]"), un div con classe "content" con il testo del messaggio (un frammento HTML), un bottone con classe "like" e testo "Mi piace", un bottone con classe "Reply" e testo "Rispondi" e un p di classe "LikeList" che contiene o niente (se nessuno ha fatto Like sul messaggio) oppure la frase "Questo messaggio piace a XX lettori", dove XX è un numero. All'inizio della lista c'è un bottone con classe "New" e testo "Crea nuovo messaggio".

Esiste una funzione "edit_message(message_id), che mostra un form per la creazione di un messaggio. Questa funzione vuole come parametro l'id del messaggio di cui questo è una risposta, o null se è un messaggio nuovo, e restituisce il testo del messaggio, ma NON NE FA IL SUBMIT. Non è necessario implementare questa funzione.

Presso il server sono disponibili tre servizi, il primo, all'URL http://www.forum.com/submit.php, accetta la sottomissione di nuovi messaggi con POST, con parametro reply_to opzionale che indica l'id del messaggio di cui questo è una risposta. Il secondo, disponibile all'URL http://www.forum.com/like.php, accetta la sottomissione di un like su uno specifico messaggio, usando il metodo POST e il parametro message_id con l'id del messaggio apprezzato. Il terzo servizio è all'URL http://www.forum.com/show.php serve per accedere in GET alla lista di messaggi disponibili in una discussione, e ha parametri "thread" che contiene l'id della discussione. Non è necessario implementare questi servizi. Tutti i servizi restituiscono, in caso di successo, un oggetto JSON della forma:

```
{
        thread_id: 'thread_12345',
        version: 1,
        timestamp: 12345,
        messages: [{
                id: 'message_12345',
                like: '25',
                user: 'user_23456',
                userName: 'Rosie',
                content: '<p>Dr. Who è il miglior telefilm inglese della storia.</p>'
                }, {
                id: 'message_12346',
                like: '0',
                user: 'user_45678',
                userName: 'Dalek',
                content: '<p>Basta con il Dr. Who. Uscite di casa, fate qualcosa
                                di produttivo.</p>'
                },
                ... ]
}
```

Si realizzi usando un framework Ajax a propria scelta tra JQuery e ExtJS:

1. Una funzione, chiamata show_message(l), che visualizza la lista di messaggi l nella maniera più appropriata al framework e al proprio gusto estetico.
2. la funzione click del pulsante "Mi piace", che chiama asincronamente il servizio

like.php con l'id corretto. Alla ricezione della risposta aggiorna la visualizzazione della lista se tutto è andato bene o emette un alert con il messaggio di errore se ne è stato ricevuto uno.

3. la funzione click del pulsante "Rispondi", che chiama asincronamente la funzione edit_message() con il parametro corretto e a conclusione, se il messaggio non è vuoto, lo spedisce al servizio submit.php con i parametri corretti. Alla ricezione della risposta aggiorna la visualizzazione della lista se tutto è andato bene o emette un alert con il messaggio di errore se ne è stato ricevuto uno.
4. un servizio asincrono, che si attiva ogni 10 secondi, che richiede una lista aggiornata dei messaggi della discussione in corso. Alla ricezione della risposta aggiorna la visualizzazione della lista se tutto è andato bene o non fa niente se è stato ricevuto un errore. A tal proposito, la funzione di Javascript setTimeout(f,m) esegue tra m millisecondi la funzione f. Si noti che il timeout viene attivato una volta soltanto, e quindi va riattivato ogni volta. La funzione setTimeout() restituisce immediatamente un codice identificativo del timeout che può essere usato come parametro della funzione clearTimeout(t) che cancella la chiamata differita.

5. (+1 punto) Per risparmiare costi di trasmissione su discussioni lunghe, la versione 2 dei servizi (come indicato dal valore 'version' del JSON di risposta) omette i blocchi relativi a messaggi invariati, mentre spedisce il blocco JSON come sopra (cioè con content e indicazione dell'utente) per i soli messaggi nuovi, un blocco come:

   ```
   {
           id: 'message_12346',
           like: '5'
   }
   ```

   per i messaggi che hanno cambiato solo il numero di Like, e un blocco come:

   ```
   {
           id: 'message_12346'
   }
   ```

   per i messaggi rimossi dall'ultima spedizione della lista (e al suo posto va visualizzato un testo tipo "Messaggio rimosso dal moderatore della discussione"). Si cambi la funzione show_message(l) per gestire anche la versione 2 di questi servizi.

6. (+ 1 punto) Si realizzi la funzione edit_message(message_id) che visualizza un form di inserimento del messaggio vicino al messaggio passato come parametro, oppure vicino al pulsante "New" altrimenti.

Semantic Web (6 punti)

Considerate la seguente descrizione in linguaggio naturale:

I libri "Il ragazzo del cimitero" e "Nessundove" di Neil Gaiman sono stati pubblicati rispettivamente da Mondadori (nel 2009) e da Fannucci (nel 1995). Nel 2005, Mondadori ha anche pubblicato "Gomorra" di Roberto Saviano, mentre Feltrinelli ha presentato "Margherita Dolcevita" di Stefano Benni.

Rispondere ai seguenti quesiti:

1. In un formato a scelta tra RDF/XML e Turtle, descrivere in RDF la descrizione presentata. I tipi (proprietà *rdf:type*) delle varie risorse da descrivere possono essere: *foaf:Person* per descrivere le persone, *foaf:Organization* per descrivere gli editori e *foaf:Document* per descrivere i libri. Inoltre, utilizzare le proprietà *foaf:name* per specificare il nome delle persone e delle organizzazioni, *dcterms:title* e *dcterms:date* per specificare il titolo e la data di pubblicazione di un libro, *dcterms:creator* e *dcterms:publisher* per collegare il libro rispettivamente con il suo autore ed editore.

2. Scrivere delle query SPARQL in modo da:

    - restituire tutte le persone che sono autori di almeno due libri;
    - restituire tutti gli autori che hanno pubblicato un libro per Feltrinelli;
    - restituire i primi autori, ordinati per nome, che abbiano pubblicato o per Mondadori o per Fannucci.

3. Considerate la descrizione precedente come racchiusa da un elemento "p" di HTML. Aggiungendo un numero appropriato di elementi HTML all'interno di *p*, usare RDFa per inserire tutte le triple RDF introdotte nel punto 1.

Teoria (4 punti)

Descrivere REST e indicare nel dettaglio i punti fondamentali della *Uniform Interface* che separa client e server.