Indice

Introduzione al Web

Il concetto di ipertesto, WWW e Internet

- **Internet** è l'infrastruttura tecnologica (rete fisica) che permette la comunicazione tra computer
- **World Wide Web** è un sistema software che funziona su Internet per accedere a documenti collegati
- **Internet** = INTERconnected NETworks, reti interconnesse che utilizzano TCP/IP
- Internet è nato nel 1969 con ARPAnet e si è evoluto negli anni '80-'90

Il World Wide Web Consortium (W3C)

- Organismo indipendente che definisce gli standard web
- Membri: principali aziende tecnologiche e università
- Propone standard che diventano Recommendation dopo un processo di revisione
- Offre: definizione dello standard, test suite e servizi di validazione

HyperText Transfer Protocol (HTTP)

- Protocollo di comunicazione client-server per lo scambio di documenti web
- Nato al CERN da Tim Berners-Lee (1990)
- Metodi: GET, POST, HEAD, PUT, DELETE
- Codici di stato: 1xx (informativo), 2xx (successo), 3xx (reindirizzamento), 4xx (errore client), 5xx (errore server)

HTML

Storia e Evoluzione

- HTML (HyperText Markup Language): linguaggio di markup per creare pagine web
- Evoluzione: HTML → XHTML → HTML5
- XHTML: riformulazione di HTML come XML (più rigido, regole più severe)
- HTML5: standard attuale, più flessibile, introduce nuovi elementi semantici

Sintassi di base HTML/XHTML

XHTML - Regole sintattiche

- Tag e attributi case sensitive (minuscolo)
- Tag devono essere sempre chiusi (`<br />`)
- Nidificazione corretta dei tag
- Valori degli attributi tra virgolette doppie
- Tutti gli attributi devono avere un valore

HTML5 - Semplificazioni

- Sintassi più flessibile
- DOCTYPE semplificato: `<!DOCTYPE html>`
- Tag `<meta charset="utf-8">` semplificato
- Non necessari i type per script e CSS

Struttura del documento

```
<!DOCTYPE html>
<html lang="it">
<head>
    <meta charset="utf-8">
    <title>Titolo pagina</title>
    <meta name="description" content="Descrizione">
    <link rel="stylesheet" href="stile.css">
</head>
<body>
    <header>...</header>
    <nav>...</nav>
    <main>...</main>
    <footer>...</footer>
</body>
</html>
```

Tag Principali

Markup Semantico (HTML5)

- `<header>`: intestazione di pagina o sezione
- `<nav>`: sezione di navigazione
- `<main>`: contenuto principale
- `<article>`: contenuto autonomo e indipendente
- `<section>`: raggruppamento tematico di contenuti
- `<aside>`: contenuto correlato ma separabile
- `<footer>`: piè di pagina o conclusione sezione

Testo

- `<h1>` - `<h6>`: titoli e sottotitoli
- `<p>`: paragrafo
- `<em>`, `<strong>`: enfasi e forte enfasi
- `<ul>`, `<ol>`, `<li>`: liste non ordinate, ordinate, elementi
- `<dl>`, `<dt>`, `<dd>`: liste di definizioni
- `<blockquote>`, `<q>`, `<cite>`: citazioni

Altri elementi

- `<a href="...">`: collegamenti ipertestuali
- `<img src="..." alt="...">`: immagini
- `<figure>`, `<figcaption>`: figure con didascalie
- `<table>`, `<tr>`, `<th>`, `<td>`: tabelle
- `<div>`, `<span>`: contenitori generici

Form

```
<form action="URL" method="post">
    <fieldset>
        <legend>Titolo form</legend>
        <label for="nome">Nome:</label>
        <input type="text" id="nome" name="nome" required>

        <label for="email">Email:</label>
        <input type="email" id="email" name="email">

        <input type="submit" value="Invia">
    </fieldset>
</form>
```

Elementi form

- `<input type="...">`: campi di input (text, password, checkbox, radio, file, ecc.)
- `<textarea>`: aree di testo multi-riga
- `<select>`, `<option>`: menu a discesa
- `<button>`: pulsanti
- `<label>`: etichette associate a campi form
- `<fieldset>`, `<legend>`: raggruppamento campi

Attributi form HTML5

- `required`: campo obbligatorio
- `pattern`: valida con espressione regolare
- `placeholder`: suggerimento nel campo
- `autofocus`: focus automatico
- `min`, `max`, `step`: per campi numerici

HTML5 - Novità

- Elementi semantici
- Supporto nativo audio/video: `<audio>`, `<video>`
- Canvas per grafica: `<canvas>`
- Web Storage: localStorage, sessionStorage
- Drag and drop API
- Geolocation API
- WebWorkers per multithreading
- Form avanzati con validazione

CSS

Introduzione e concetti di base

- CSS (Cascading Style Sheets): linguaggio per definire l'aspetto delle pagine web
- Separazione tra struttura (HTML) e presentazione (CSS)
- Versioni: CSS1 → CSS2 → CSS3 (moduli)

Sintassi CSS

```
selettore {
    proprietà: valore;
```

```
    proprietà: valore;
}
```

Collegamento CSS-HTML

1. **CSS Esterno** (metodo preferito)

```
<link rel="stylesheet" href="stile.css" type="text/css">
```

2. **CSS Interno** (nell'head)

```
<style type="text/css">
    p { color: red; }
</style>
```

3. **CSS Inline** (attributo style)

```
<p style="color: red;">Testo rosso</p>
```

Selettori CSS

- **Selettori di tipo**: `p { ... }`
- **Selettori ID**: `#idname { ... }`
- **Selettori classe**: `.classname { ... }`
- **Selettori discendenti**: `div p { ... }`
- **Selettori figli**: `ul > li { ... }`
- **Selettori di attributo**: `a[href="..."] { ... }`
- **Pseudoclassi**: `a:hover { ... }`
- **Pseudoelementi**: `p::first-letter { ... }`

Specificità

- Calcolo: (ID, Classi+Attributi, Elementi)
- Esempio: `#nav a` = (1,0,1), `a` = (0,0,1)
- Regola: maggiore specificità vince

Cascata e ereditarietà

Ordine di applicazione:

1. Impostazioni predefinite browser
2. Fogli di stile esterni

3. Fogli di stile interni (head)
4. Stile inline

Importanza:

- `!important` sovrascrive le normali regole

Box Model

```
+-----------------------+
| Margin                |
| +-----------------+   |
| | Border          |   |
| | +-----------+   |   |
| | | Padding   |   |   |
| | | +-------+ |   |   |
| | | |Content| |   |   |
| | | +-------+ |   |   |
| | +-----------+   |   |
| +-----------------+   |
+-----------------------+
```

- **Content**: contenuto effettivo
- **Padding**: spazio tra contenuto e bordo
- **Border**: bordo attorno al padding
- **Margin**: spazio esterno al bordo

Layout CSS

Posizionamento

- `position: static` (normale flusso)
- `position: relative` (rispetto alla posizione normale)
- `position: absolute` (rispetto all'antenato posizionato)
- `position: fixed` (rispetto alla viewport)
- `position: sticky` (ibrido tra relative e fixed)

Float

- `float: left` o `float: right`
- `clear: both`, `clear: left`, `clear: right`

Flexbox

```css
.container {
    display: flex;
    flex-direction: row;
    justify-content: space-between;
    align-items: center;
}
```

Grid

```css
.container {
    display: grid;
    grid-template-columns: 1fr 2fr 1fr;
    grid-gap: 20px;
}
```

Responsive Web Design

- Media Query: `@media screen and (max-width: 768px) { ... }`
- Layout fluidi o elastici
- Unità relative: em, rem, %, vh, vw
- Immagini responsive

CSS3 - Novità

- Transizioni e animazioni
- Trasformazioni (rotate, scale, translate)
- Ombreggiature (box-shadow, text-shadow)
- Border-radius
- Multiple backgrounds
- Gradients
- Variabili CSS
- Filtri e effetti

JavaScript

Introduzione

- Linguaggio di scripting client-side
- Creato nel 1995 da Netscape (LiveScript)
- Non è un sottoinsieme di Java nonostante il nome
- Permette di creare pagine web dinamiche e interattive

Inserimento in una pagina HTML

```
<!-- Script esterno -->
<script src="script.js"></script>

<!-- Script interno -->
<script>
    // Codice JavaScript
</script>

<!-- Script inline (evento) -->
<button onclick="alert('Ciao!')">Clicca</button>
```

Sintassi di base

```
// Variabili
var nome = "Mario";
let età = 30;
const PI = 3.14;

// Funzioni
function saluta(nome) {
    return "Ciao " + nome;
}

// Condizionali
if (età >= 18) {
    console.log("Maggiorenne");
} else {
    console.log("Minorenne");
}

// Cicli
for (let i = 0; i < 5; i++) {
    console.log(i);
}

while (condizione) {
    // codice
}
```

Tipi di dati

- **Number**: numeri interi e decimali
- **String**: testo tra virgolette
- **Boolean**: true o false

- **Object**: collezioni di proprietà
- **Array**: liste ordinate
- **null/undefined**: valori speciali

Oggetti e Array

```
// Oggetto
let persona = {
    nome: "Mario",
    cognome: "Rossi",
    età: 30,
    saluta: function() {
        return "Ciao, sono " + this.nome;
    }
};

// Array
let numeri = [1, 2, 3, 4, 5];
numeri.push(6);  // aggiunge elemento
numeri.pop();    // rimuove ultimo elemento
```

DOM (Document Object Model)

- Rappresentazione ad albero della struttura HTML
- Permette di manipolare dinamicamente il contenuto

```
// Selezione elementi
let elemento = document.getElementById("id");
let elementi = document.getElementsByTagName("p");
let classi = document.getElementsByClassName("classe");
let selettori = document.querySelectorAll("div.classe");

// Modifica contenuto
elemento.innerHTML = "Nuovo contenuto";
elemento.style.color = "red";

// Aggiunta elementi
let nuovoEl = document.createElement("div");
nuovoEl.textContent = "Nuovo elemento";
document.body.appendChild(nuovoEl);

// Rimozione elementi
elemento.parentNode.removeChild(elemento);
```

Eventi

```
// Modello tradizionale
elemento.onclick = function() {
    // codice
};

// Event listener
elemento.addEventListener("click", function(event) {
    // codice
});

// Rimozione
elemento.removeEventListener("click", nomeFunzione);
```

Tipi di eventi comuni:

- `click`, `dblclick`
- `mouseenter`, `mouseleave`
- `keydown`, `keyup`
- `submit`, `change`, `input`
- `load`, `resize`, `scroll`

AJAX (Asynchronous JavaScript And XML)

```
// XMLHttpRequest
let xhr = new XMLHttpRequest();
xhr.onreadystatechange = function() {
    if (xhr.readyState === 4 && xhr.status === 200) {
        console.log(xhr.responseText);
    }
};
xhr.open("GET", "url", true);
xhr.send();

// Fetch API (moderno)
fetch("url")
    .then(response => response.json())
    .then(data => console.log(data))
    .catch(error => console.error(error));
```

JavaScript moderno (ES6+)

- Arrow functions: `() => { }`
- Template literals: `` `Ciao ${nome}` ``
- Destructuring: `let {nome, età} = persona`
- Spread operator: `[...array1, ...array2]`

- Promise e async/await
- Modules (import/export)
- Classes

PHP

Introduzione a PHP

- PHP: PHP Hypertext Processor (acronimo ricorsivo)
- Linguaggio di scripting server-side
- Creato nel 1994, attualmente alla versione 8
- Utilizzato per generare pagine dinamiche
- Interagisce con database (es. MySQL)

Sintassi di base

```php
<?php
    // Commento su una linea
    /* Commento
       su più
       linee */

    // Stampa
    echo "Hello World!";
    print("Hello World!");

    // Variabili
    $nome = "Mario";
    $età = 30;

    // Concatenazione
    echo "Ciao " . $nome . ", hai " . $età . " anni.";
?>
```

Tipi di dati

- **Scalari**: boolean, integer, float, string
- **Composti**: array, object
- **Speciali**: resource, NULL

Operatori

- Aritmetici: `+`, `-`, `*`, `/`, `%`, `**`

- Assegnazione: `=`, `+=`, `-=`, etc.
- Confronto: `==`, `===`, `!=`, `<>`, `!==`, `<`, `>`, etc.
- Logici: `&&`, `||`, `!`, `and`, `or`, `xor`
- Incremento/decremento: `++`, `--`

Strutture di controllo

```php
// Condizionali
if ($condizione) {
    // codice
} elseif ($altraCondizione) {
    // codice
} else {
    // codice
}

switch ($variabile) {
    case 'valore1':
        // codice
        break;
    case 'valore2':
        // codice
        break;
    default:
        // codice
}

// Cicli
for ($i = 0; $i < 10; $i++) {
    // codice
}

while ($condizione) {
    // codice
}

do {
    // codice
} while ($condizione);

foreach ($array as $valore) {
    // codice
}

foreach ($array as $chiave => $valore) {
    // codice
}
```

Array

```php
// Array indicizzato
$numeri = array(1, 2, 3, 4, 5);
$numeri = [1, 2, 3, 4, 5];  // sintassi breve

// Array associativo
$persona = array(
    "nome" => "Mario",
    "cognome" => "Rossi",
    "età" => 30
);

// Accesso
echo $numeri[0];         // 1
echo $persona["nome"];   // Mario

// Funzioni array
count($array);           // lunghezza
array_push($array, $elemento);  // aggiunge elemento
array_pop($array);       // rimuove ultimo elemento
sort($array);            // ordina
implode(", ", $array);   // unisce elementi con separatore
```

Funzioni

```php
function saluta($nome, $cognome = "sconosciuto") {
    return "Ciao $nome $cognome!";
}

// Chiamata
$messaggio = saluta("Mario", "Rossi");  // Ciao Mario Rossi!
$messaggio = saluta("Mario");           // Ciao Mario sconosciuto!

// Passaggio per referenza
function incrementa(&$numero) {
    $numero++;
}
```

Gestione form

```php
// Accesso ai dati
$nome = $_GET['nome'];       // metodo GET
$email = $_POST['email'];    // metodo POST
$qualsiasi = $_REQUEST['campo'];  // GET o POST

// Validazione
```

```php
if (isset($_POST['submit'])) {
    $email = filter_input(INPUT_POST, 'email', FILTER_VALIDATE_EMAIL);
    if ($email === false) {
        echo "Email non valida";
    }
}
```

Sessioni e Cookie

```php
// Sessioni
session_start();
$_SESSION['user'] = "Mario";
echo $_SESSION['user'];
unset($_SESSION['user']);  // rimuove variabile
session_destroy();         // distrugge sessione

// Cookie
setcookie("nome", "valore", time() + 3600);  // valido per 1 ora
echo $_COOKIE['nome'];
```

Connessione al Database (MySQLi)

```php
// Connessione
$conn = new mysqli("localhost", "username", "password", "database");

// Verifica connessione
if ($conn->connect_error) {
    die("Connessione fallita: " . $conn->connect_error);
}

// Query
$sql = "SELECT id, nome, cognome FROM utenti";
$result = $conn->query($sql);

// Recupero dati
if ($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
        echo "ID: " . $row["id"] . " - Nome: " . $row["nome"] . "<br>";
    }
} else {
    echo "0 risultati";
}

// Prepared statement (sicuro)
$stmt = $conn->prepare("INSERT INTO utenti (nome, email) VALUES (?, ?)");
$stmt->bind_param("ss", $nome, $email);  // "ss" = due stringhe
$stmt->execute();
```

```php
// Chiusura
$stmt->close();
$conn->close();
```

Sicurezza

- **SQL Injection**: usare prepared statements
- **XSS (Cross-Site Scripting)**: filtrare input con `htmlspecialchars()`
- **CSRF (Cross-Site Request Forgery)**: token di verifica
- **Validazione input**: `filter_input()`, `filter_var()`
- **Password**: `password_hash()` e `password_verify()`

Accessibilità

Definizione e Importanza

- **Accessibilità web**: capacità di un sito di essere fruibile da tutti, incluse persone con disabilità
- **Universal Design**: progettazione per tutti, indipendentemente dalle abilità
- Categorie di utenti: disabilità visive, uditive, motorie, cognitive

Normative e Standard

- **WCAG (Web Content Accessibility Guidelines)**: linee guida W3C
  - Versioni: WCAG 1.0 (1999), WCAG 2.0 (2008), WCAG 2.1 (2018), WCAG 2.2 (2023)
  - Livelli: A (base), AA (intermedio), AAA (avanzato)
- **Legge Stanca** (Legge 4/2004): normativa italiana sull'accessibilità
  - Obbliga le PA ad avere siti accessibili
  - AGID monitora l'attuazione
- **Accessibility Act** (Direttiva UE 2019/882)
  - Estende gli obblighi ai privati entro il 2025

I 4 Principi WCAG

1. **Percepibile**: informazioni presentabili in forme percepibili da tutti
2. **Utilizzabile**: componenti UI e navigazione utilizzabili da tutti
3. **Comprensibile**: informazioni e funzionamento comprensibili
4. **Robusto**: contenuto interpretabile da vari user agent

Tecniche per garantire l'accessibilità

Testo e struttura

- Markup semantico corretto
- Intestazioni gerarchiche (h1-h6)
- Liste ordinate e non ordinate
- Contrasto adeguato (4.5:1 minimo)
- Dimensioni font scalabili
- Evitare testo scorrevole o lampeggiante

Immagini e media

- Attributo `alt` per le immagini
- Trascrizioni per audio
- Sottotitoli per video
- Evitare GIF animate con lampeggiamenti rapidi

Link e navigazione

- Link descrittivi (evitare "clicca qui")
- Skip link per saltare la navigazione
- Accesskey e tabindex
- Breadcrumb e wayfinding

Form e interazione

- Label associate ai campi
- Messaggi di errore chiari
- Fieldset e legend per raggruppare
- Tempo sufficiente per completare le azioni

Tabelle

- Caption e summary
- Intestazioni corrette (th con scope)
- Evitare tabelle per layout
- Struttura semplice e chiara

WAI-ARIA (Accessible Rich Internet Applications)

- Estende HTML con attributi di accessibilità
- Ruoli, stati e proprietà

- Esempi:
  - `role="navigation"`, `role="search"`
  - `aria-label`, `aria-labelledby`
  - `aria-hidden`, `aria-expanded`

Test di accessibilità

- Validazione automatica + test manuali
- Strumenti: WAVE, AXE, Lighthouse
- Test con tastiera (senza mouse)
- Test con screen reader
- Test con utenti reali

Architettura dell'Informazione

Definizione e Principi

- **Architettura dell'informazione**: organizzazione, strutturazione ed etichettatura dell'informazione
- Efficace quando aiuta gli utenti a trovare ciò che cercano

Contesto, Contenuto, Utenti

- **Contesto**: obiettivi organizzativi, vincoli, risorse
- **Contenuto**: tipologia, struttura, volume, governance
- **Utenti**: bisogni, comportamenti, preferenze

Comportamento di ricerca degli utenti

- **Metafora della pesca**:
  - **Tiro perfetto**: utenti che sanno esattamente cosa cercano
  - **Trappola per aragoste**: utenti con idea vaga che imparano durante la ricerca
  - **Pesca con la rete**: utenti che vogliono esplorare un argomento generale
  - **Boa di segnalazione**: utenti che vogliono ritrovare informazioni già viste

Schemi Organizzativi

Schemi Esatti

- **Alfabetico**: organizzazione per ordine alfabetico
- **Cronologico**: organizzazione per data/tempo

- **Geografico**: organizzazione per luogo

Caratteristiche:

- Mutualmente esclusivi
- Facili da progettare e mantenere
- Richiedono che l'utente conosca il nome specifico

Schemi Ambigui

- **Per argomento (topic)**: organizzazione per tema
- **Per attività (task)**: organizzazione per funzione/compito
- **Per audience**: organizzazione per tipo di utente
- **Metaforici**: organizzazione basata su metafore familiari

Caratteristiche:

- Più difficili da progettare
- Più utili quando l'utente non sa esattamente cosa cerca
- Permettono serendipità e apprendimento associativo

Strutture Organizzative

- **Gerarchia**: struttura ad albero (padre-figlio)
  - Ampiezza vs profondità (preferire ampie e poco profonde)
  - Legge di Hick: tempo decisionale aumenta con opzioni
- **Sequenza**: percorso lineare
  - Utile per tutorial, processi step-by-step
- **Ipertesto**: collegamenti non gerarchici
  - Flessibili ma possono disorientare
- **Faccette/Matrici**: multiple dimensioni per filtrare
  - Utili per grandi database (e-commerce)

Sistemi di navigazione

- **Globale**: navigazione principale (menu)
- **Locale**: navigazione specifica della sezione
- **Contestuale**: link in-page correlati
- **Supplementare**: sitemap, indici, guide

Etichettatura

- Creare etichette chiare e consistenti
- Evitare gergo e ambiguità

- Usare il linguaggio degli utenti
- Testare la comprensibilità

SEO

Definizione e Importanza

- **SEO (Search Engine Optimization)**: ottimizzare un sito per i motori di ricerca
- Obiettivo: migliorare posizionamento nelle SERP (Search Engine Results Page)
- Traffico organico vs traffico a pagamento

Fasi dell'ottimizzazione

1. **Ottimizzazione tecnica**: permettere ai motori di accedere e indicizzare
2. **Creazione contenuti**: contenuti rilevanti e di qualità
3. **Promozione**: raccolta di link da siti autorevoli

Fattori di ranking

Fattori interni (On-Page)

- **Keyword**:
  - Nel tag title (max 55 caratteri)
  - Nel meta description (max 145 caratteri)
  - Nelle intestazioni (h1, h2, etc.)
  - Nel corpo del testo (naturale, non keyword stuffing)
- **Contenuto**:
  - Qualità e originalità
  - Lunghezza adeguata
  - Aggiornamento regolare
  - Struttura chiara (intestazioni, paragrafi)
- **Markup tecnico**:
  - HTML valido e semantico
  - Schema.org e dati strutturati
  - URL SEO-friendly
  - Sitemap XML
- **Velocità e performance**:
  - Tempo di caricamento
  - Ottimizzazione immagini
  - Minificazione CSS/JS

    - Mobile-friendly

Fattori esterni (Off-Page)

- **Backlink** (link in entrata)
    - Qualità > quantità
    - Autorevolezza dei siti linkanti
    - Pertinenza tematica
    - Testo dell'ancora
- **Social signals**
    - Condivisioni sui social
    - Menzioni del brand
- **Comportamento utente**
    - Tempo di permanenza
    - Tasso di rimbalzo
    - CTR (Click-Through Rate)

Strumenti SEO

- Google Search Console
- Google Analytics
- Strumenti di analisi keyword
- Tool di audit SEO

Best Practices

- Ricerca delle keyword
- Contenuti per rispondere all'intento di ricerca
- Ottimizzazione immagini (alt text, dimensioni)
- Internal linking strategico
- URL descrittivi e brevi
- Evitare contenuti duplicati
- Mobile-first design

UX e Design

User Experience (UX)

- **UX**: esperienza complessiva dell'utente nell'interazione con un prodotto/servizio
- **UI (User Interface)**: aspetto visivo e interattivo dell'interfaccia

- **Usabilità**: facilità d'uso di un sistema (efficacia, efficienza, soddisfazione)

Principi di Web Design

Layout e Struttura

- **Area visibile (above the fold)**: contenuto visibile senza scroll
- **Griglia**: organizzazione ordinata degli elementi
- **Spazio bianco**: migliora leggibilità e focus
- **Contrasto**: evidenzia elementi importanti
- **Gerarchia visiva**: guida l'attenzione dell'utente

Tipografia

- Leggibilità (font size, line-height)
- Contrasto testo-sfondo
- Gerarchia tipografica
- Font web-safe o webfonts

Colore

- **Palette coerente**: 2-3 colori primari + accenti
- **Significato dei colori**: connotazioni culturali
- **Contrasto**: accessibilità (4.5:1 minimo)
- **Feedback**: stati interattivi (hover, focus)

Responsive Web Design

- **Mobile First**: progettare prima per mobile
- **Media Query**: adattamento a diverse dimensioni schermo
- **Breakpoint**: punti di cambiamento layout
  - 320px: smartphone portrait
  - 768px: tablet portrait
  - 1024px: tablet landscape/desktop
  - 1200px: desktop large
- **Layout fluidi**: dimensioni relative (%, em, rem)
- **Immagini responsive**: max-width, picture element

Emotional Design

- **Livelli** (Don Norman):
  - Viscerale (aspetto)
  - Comportamentale (funzionamento)

    - Riflessivo (significato)
- **Emozioni utilizzabili**:
    - Sorpresa
    - Piacere
    - Anticipazione
    - Status/Esclusività
    - Rewards

Convenzioni e pattern

- **Convenzioni esterne**: standard web comuni
- **Convenzioni interne**: coerenza all'interno del sito
- **Pattern di design**: soluzioni collaudate per problemi comuni
    - Navigation (menu, breadcrumb)
    - Forms (step, validazione)
    - Search
    - Pagination
    - Modals/popups

Feedback e Feedforward

- **Feedback**: informazione di ritorno dopo un'azione
- **Feedforward**: anticipazione dell'effetto di un'azione

Progettazione

Processo di progettazione web

1. **Analisi dei requisiti**:
    - Identificazione target utenti
    - Definizione obiettivi
    - Analisi competitor
2. **Architettura informativa**:
    - Organizzazione contenuti
    - Struttura navigazione
    - Wireframing
3. **Visual design**:
    - Stile grafico
    - Colori e tipografia
    - Immagini e icone

4. **Sviluppo**:
    - Front-end (HTML, CSS, JS)
    - Back-end (PHP, database)
    - Integrazione CMS
5. **Testing**:
    - Funzionalità
    - Usabilità
    - Accessibilità
    - Performance
6. **Lancio e manutenzione**

Team di sviluppo web

- **Project Manager**: coordinamento
- **Information Designer**: strutturazione dei contenuti
- **Web Designer**: aspetto grafico
- **Sviluppatore front-end**: implementazione interfaccia
- **Sviluppatore back-end**: funzionalità server-side
- **UX Designer**: esperienza utente
- **Content Manager**: gestione contenuti
- **SEO Specialist**: ottimizzazione motori di ricerca

Internazionalizzazione e Localizzazione

- **Internazionalizzazione**: progettare per uso globale
    - UTF-8 encoding
    - Testo espandibile
    - Layout flessibili
    - Indipendenza culturale
- **Localizzazione**: adattamento per mercati specifici
    - Traduzione
    - Adattamento culturale
    - Formati data/ora/valuta
    - Contenuti specifici per regione

Differenze:

- Internazionalizzazione: stesso sito usabile ovunque
- Localizzazione: versioni diverse per paesi/culture

Modello di progetto web

- **Requisiti**: scopo, target, funzionalità
- **Risorse**: budget, tempo, competenze
- **Deliverable**: documenti, prototipi, prodotto finito
- **Milestones**: punti di controllo del progetto
- **Stakeholder**: client, utenti, team

Consigli per l'esame

- Studiare tutti gli argomenti in modo uniforme
- Prestare attenzione alle definizioni precise (schemi organizzativi, strutture organizzative, etc.)
- Esercitarsi su:
  - Tabelle accessibili
  - Specificità CSS
  - Domande V/F con motivazione

Bibliografia e Risorse

- W3C
- HTML5 Doctor
- CSS-Tricks
- MDN Web Docs
- A List Apart
- Nielsen Norman Group
- WCAG 2.1
- AGID