AGID
Agenzia per l'Italia Digitale

FormezPA

FORMAZIONE AGID – FORMEZ SULLA TRANSIZIONE DIGITALE DELLA PA

Progetto Informazione e formazione per la transizione digitale della PA nell'ambito del progetto «Italia Login – la casa del cittadino»

(A valere sul PON Governance e Capacità Istituzionale 2014-2020)

UNIONE EUROPEA
Dipartimento della Funzione Pubblica
AGID Agenzia per l'Italia digitale

AGID
Agenzia per l'Italia Digitale

FormezPA

Accessibilità dei siti web

1 ottobre 2021

Fabrizio Caccavello

UNIONE EUROPEA

Dipartimento della Funzione Pubblica

AGID Agenzia per l'Italia digitale

Accessibilità: errori di implementazione più comuni

Fabrizio Caccavello

UNIONE EUROPEA
Dipartimento della Funzione Pubblica
AGID Agenzia per l'Italia digitale
FormezPA

- Cosa vedremo

- Anteprima: come nasce questo tema formativo.
- Prerequisiti:
  - I framework.
  - Gli standard.
  - I sistemi automatizzati per la verifica.
- Errori comuni:
  - Testi alternativi per le immagini.
  - Titolazioni.
  - Colore e contrasto minimo.
  - Focus da tastiera.
  - Etichette.
  - Lingua della pagina.
  - Il ruolo degli elementi.

- Anteprima: come nasce questo tema formativo

Talvolta si forniscono informazioni e strategie per analizzare nei sistemi digitali in modo preliminare e rapido la coerenza con i requisiti di accessibilità (senza la pretesa di essere esaustivi rispetto alla reale conformità).

Analogamente si possono **fornire elementi di base** su come implementare interfacce utente accessibili, partendo dall'utilizzo di framework o da esempi forniti dal W3C.

Prerequisiti

- Prerequisito: i framework

L’errore primario e comune è credere che utilizzando un framework che si definisce accessibile, si possano produrre sistemi digitali accessibili.

I framework **possono essere un valido supporto** per lo sviluppo accessibile, se includono almeno due concetti fondamentali:

- **Accessibility by default**;
- **informazioni a supporto per l’implementazione**.

- Prerequisito: i framework (esempio di informazioni a supporto 1 di 2)

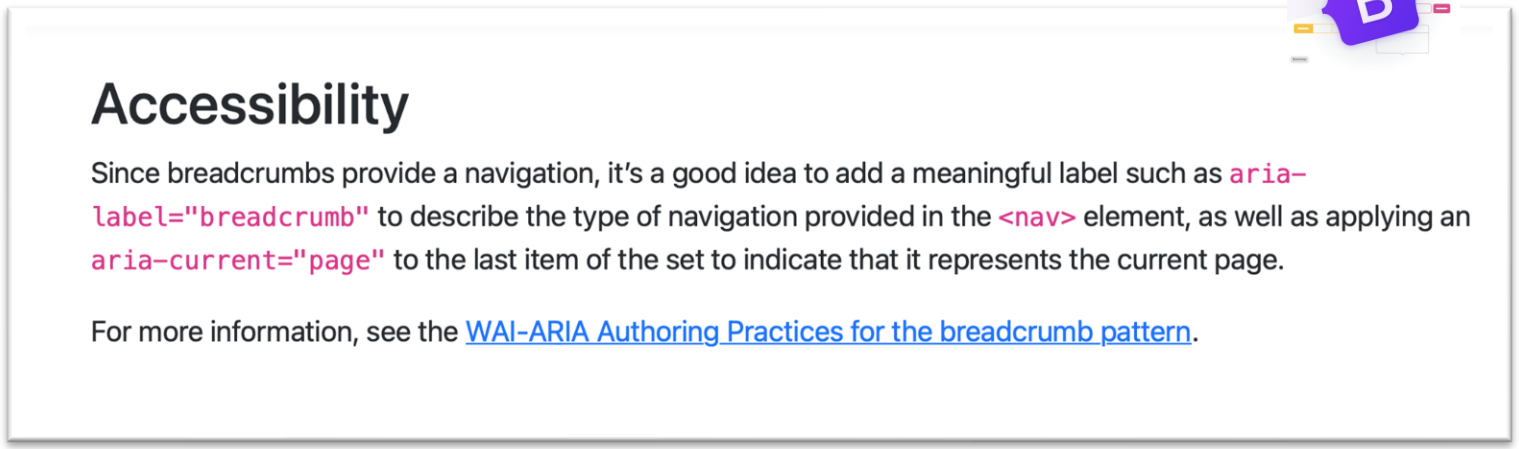
In **Bootstrap Italia ci sono molti riferimenti e informazioni** per lo sviluppo accessibile.

Esempio: la pagina delle finestre modali contiene questo tipo di informazioni

- Prerequisito: i framework (esempio di informazioni a supporto 2 di 2)

Anche **Bootstrap** contiene ampi riferimenti e supporto.

Ad esempio la pagina del componente **Breadcrumbs** contiene un chiarimento sugli **attributi necessari** per l'implementazione accessibile.



Accessibility

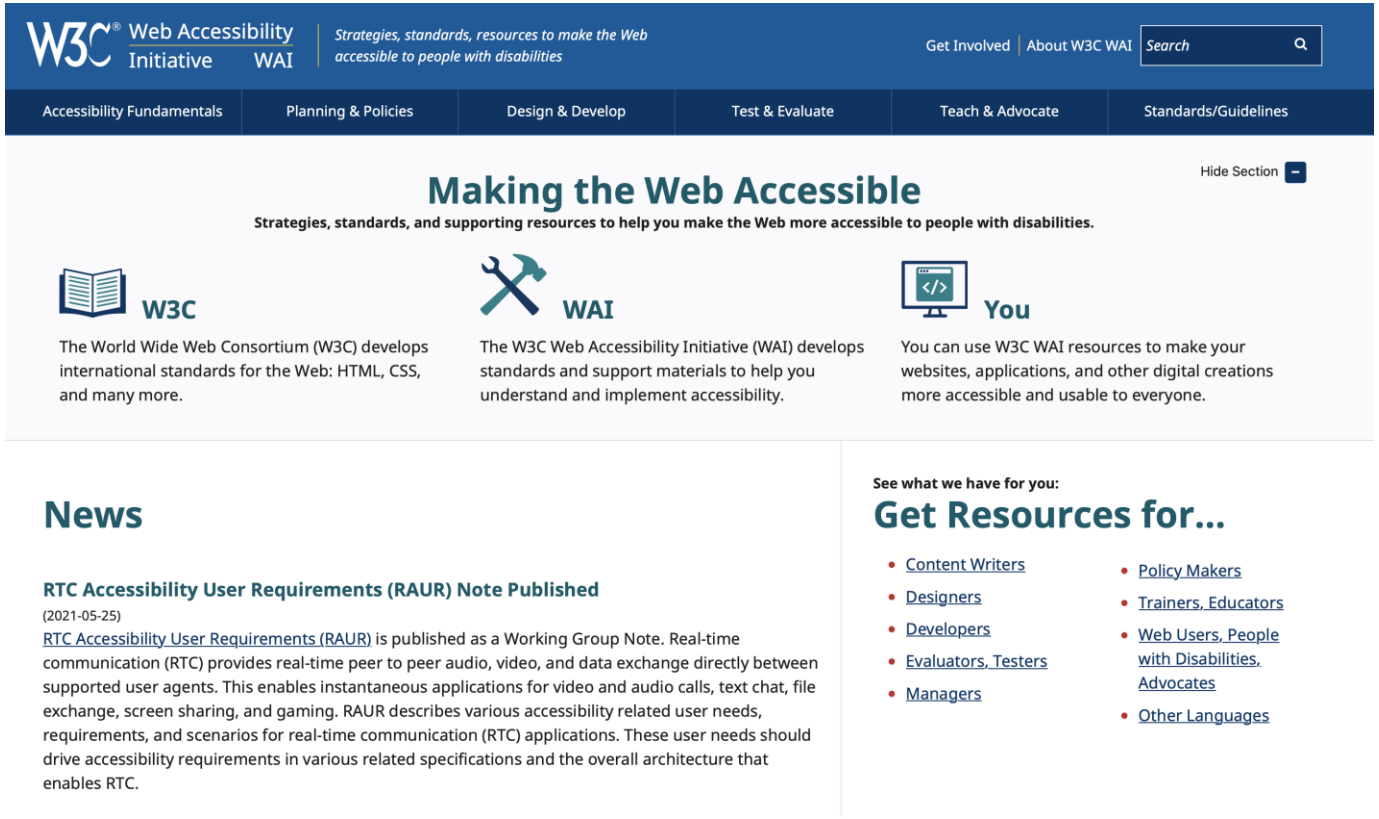
Since breadcrumbs provide a navigation, it's a good idea to add a meaningful label such as `aria-label="breadcrumb"` to describe the type of navigation provided in the `<nav>` element, as well as applying an `aria-current="page"` to the last item of the set to indicate that it represents the current page.

For more information, see the WAI-ARIA Authoring Practices for the breadcrumb pattern.

- Prerequisito: gli standard

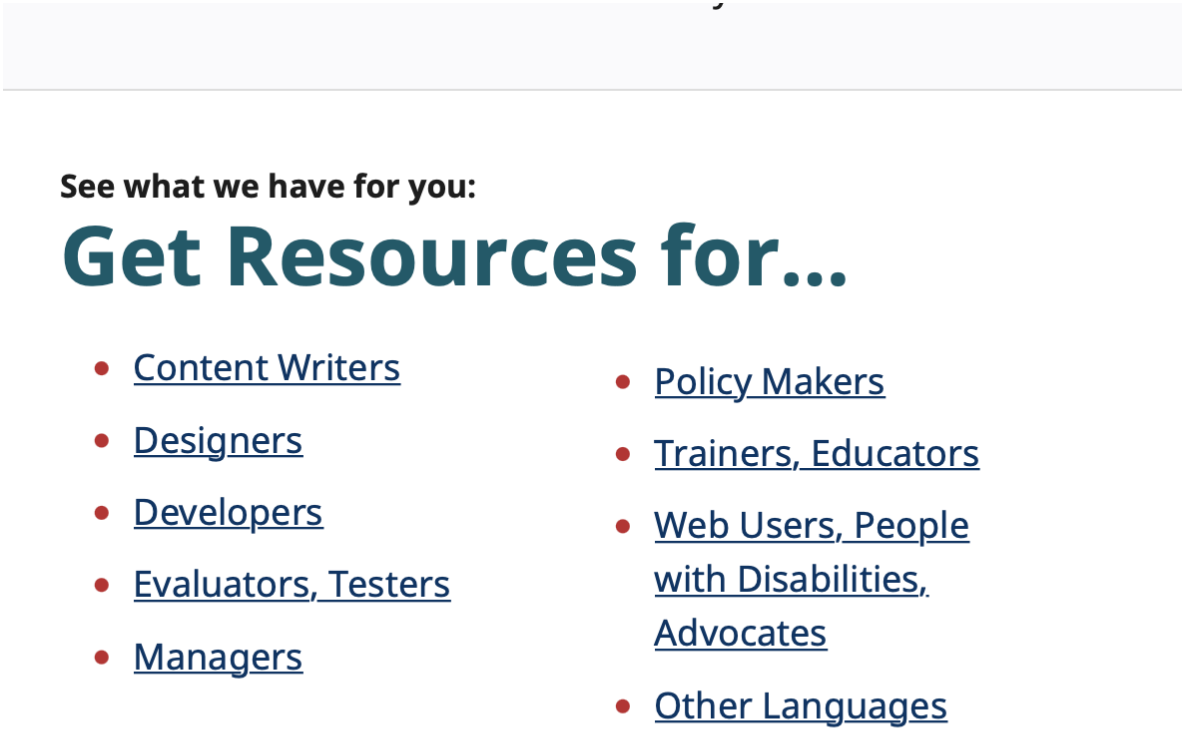
È necessario conosce bene gli standard di progettazione e i vari riferimenti: il punto di partenza è senz'altro la sezione accessibilità del W3C.
https://www.w3.org/WAI/

Le WCAG 2.1 sono l'attuale riferimento tecnico.



- Prerequisito: gli standard e i riferimenti per professionalità

Da notare come siano disponibili nella pagina riferimenti e informazioni importanti per una moltitudine di professionalità coinvolte nel processo di creazione di servizi digitali.



- Prerequisito: i sistemi automatizzati per la verifica verifica

Talvolta si pensa che i sistemi automatizzati per la verifica di accessibilità siano sufficienti per analizzare i sistemi e attestarne l'accessibilità.
Questo non è vero, anche nelle WCAG, nel capitolo introduttivo, si dice che «**le WCAG sono progettate (…) per essere verificabili tramite una combinazione di test automatici e valutazione umana**»

A tutt'oggi quindi la valutazione umana e i test automatici sono complementari per eseguire verifiche di accessibilità, e **nessun test automatico può essere sufficiente per analizzare ogni Criterio di Successo**.

Errori comuni

- Errori comuni: i testi alternativi per le immagini

Valorizzare correttamente **l'attributo alt** delle immagini, deve descrivere ciò che l'immagine stessa rappresenta nella comprensione del contenuto.

- Se **l'immagine è puramente decorativa**, cioè non è influente per la comprensione del contenuto, l'attributo **alt può essere lasciato vuoto** (`alt=" "`).
- L'attributo alt **non deve essere valorizzato con testi generici** non significativi (es: immagine, img, foto, immagine decorativa, ecc).
- L'attributo alt non deve essere valorizzato con altri contenuti testuali già presenti (es: con lo stesso testo che è stato già presentato per il titolo di un articolo).

- Errori comuni: i testi alternativi per le immagini (esempio)

Contesto: un articolo parla di una organizzazione in cui un gruppo di lavoro incontra una nuova professionista esperta in uno specifico settore e la accoglie nel gruppo di lavoro.

Testo alternativo corretto: «la coordinatrice Maria Bianchi stringe la mano alla nuova arrivata Francesca Rossi».

Testo alternativo errato: «immagine di persone al lavoro»

- Titolazioni: esempio con Bootstrap

I testi che devono rappresentare un titolo, devono esserlo sia **visivamente** che **semanticamente** (con H1, H2 ecc).

Talvolta alcune indicazioni dei framework possono indurre in errore, ad esempio quando un paragrafo ha come classe CSS uno stile che rassomiglia visivamente ad un titolo (`class="h1"`).

```
<p class="h1">h1. Bootstrap heading</p>
<p class="h2">h2. Bootstrap heading</p>
<p class="h3">h3. Bootstrap heading</p>
<p class="h4">h4. Bootstrap heading</p>
<p class="h5">h5. Bootstrap heading</p>
<p class="h6">h6. Bootstrap heading</p>
```

- Titolazioni: esempio con Bootstrap Italia

Quando il framework suggerisce che all'interno di un elemento semanticamente corretto possa esistere un elemento annidato che possa sovrascrivere lo stile originario dell'elemento padre, si può incorrere nell'errore opposto, ossia che **l'elemento possa diventare semanticamente un titolo ma visivamente un testo secondario**.

Esempio:

```
<H1>
  <span class="text-muted">
  Titolo</span>
</H1>.
```

- Colore e contrasto minimo

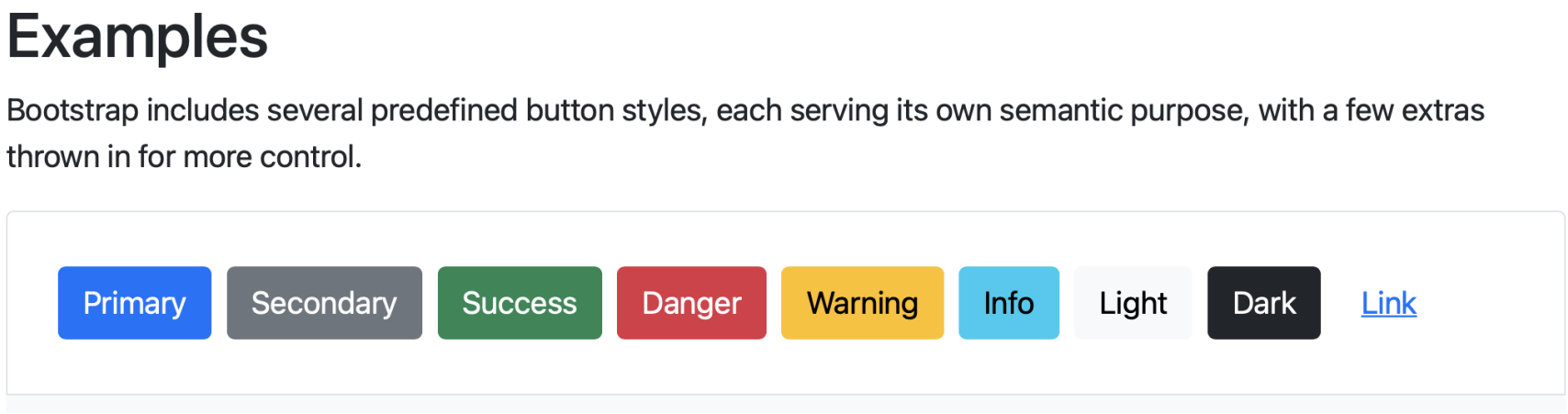
È una delle caratteristiche dell'accessibilità più note, tuttavia nelle verifiche di accessibilità è **una delle violazioni più comuni** tra quelle riscontrate.

Ci sono delle **false convinzioni** che inducono a questo errore, ad esempio che il basso contrasto possa essere elegante, o problematiche che nascono dai framework utilizzati, sia per **errate interpretazioni**, sia per **non chiare indicazioni** degli stessi framework.

- Colore e contrasto minimo: esempio positivo di Bootstrap

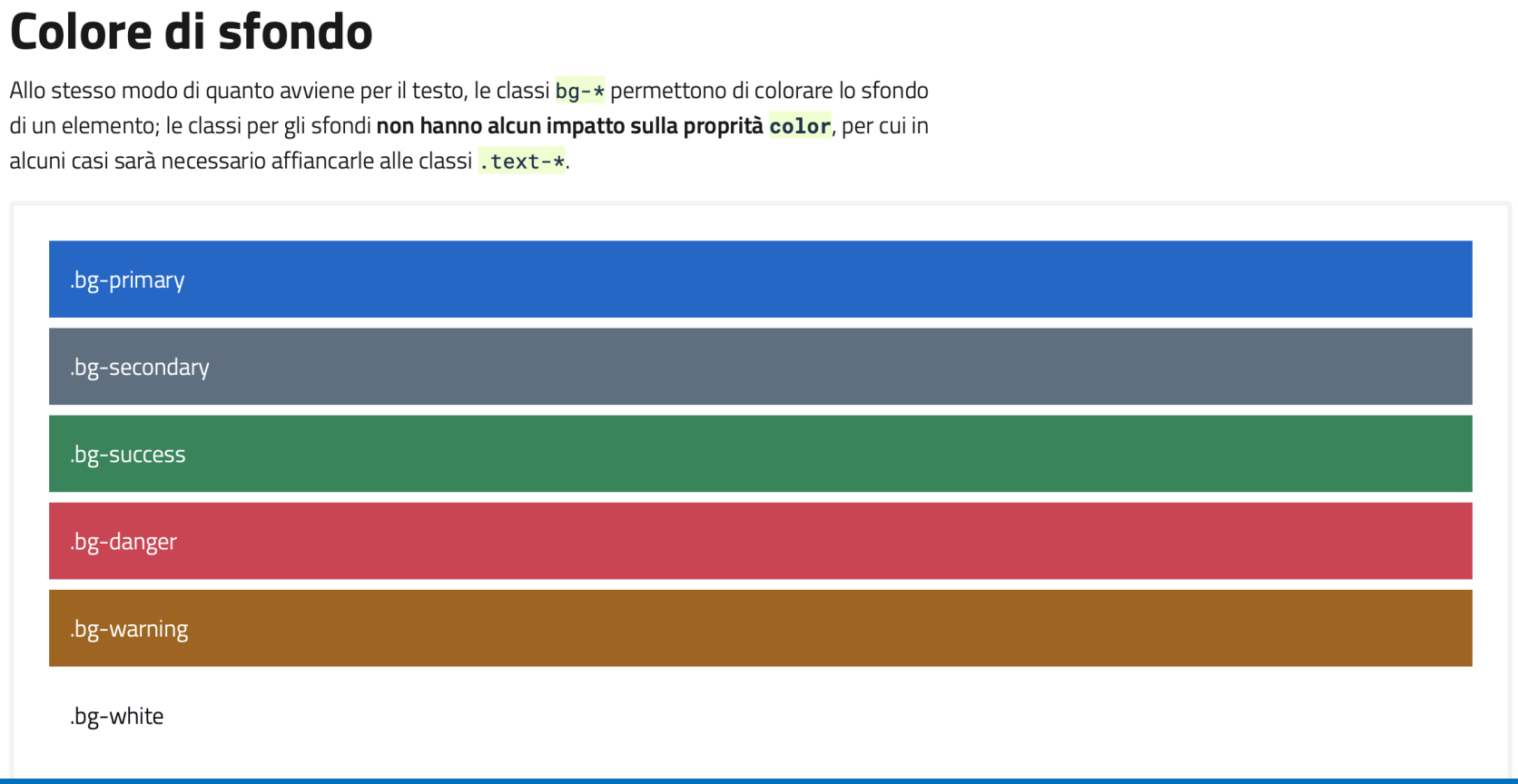
Di solito colori suggeriti nei framework forniscono un corretto contrasto tra colore primario e background, sufficiente a garantire l'accessibilità.

Esempio: schema di colori negli elementi Button di Bootstrap.



- Colore e contrasto minimo: esempio positivo di Bootstrap Italia

Esempio: schema di colori proposto dal framework Bootstrap Italia.



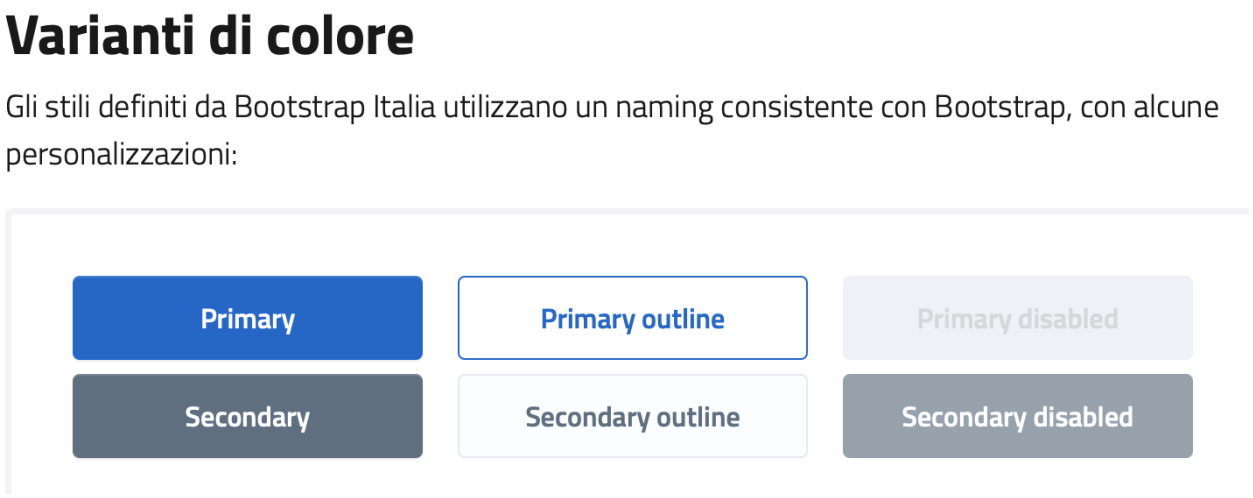
- Colore e contrasto minimo: possibili problemi con Bootstrap

In questo caso Bootstrap, a partire dalla versione 5.1, suggerisce di poter usare una nuova classe (--bs-text-opacity) impostando arbitrariamente un valore di opacità, con un suggerimento al 50%.
Non è riportato alcun alert in accessibilità ma questo utilizzo non è accessibile.

- Colore e contrasto minimo: possibili problemi con Bootstrap Italia

In **Bootstrap Italia** vengono suggerite due classi
(btn btn-primary disabled; btn btn-secondary disabled)
che permetto di impostare i pulsanti quando sono disabilitati, ma lo stato di disabilitato deve essere impostato anche semanticamente (disabled).

Se tali classi vengono usate anche per pulsanti non disabilitati si ha un errore di accessibilità.

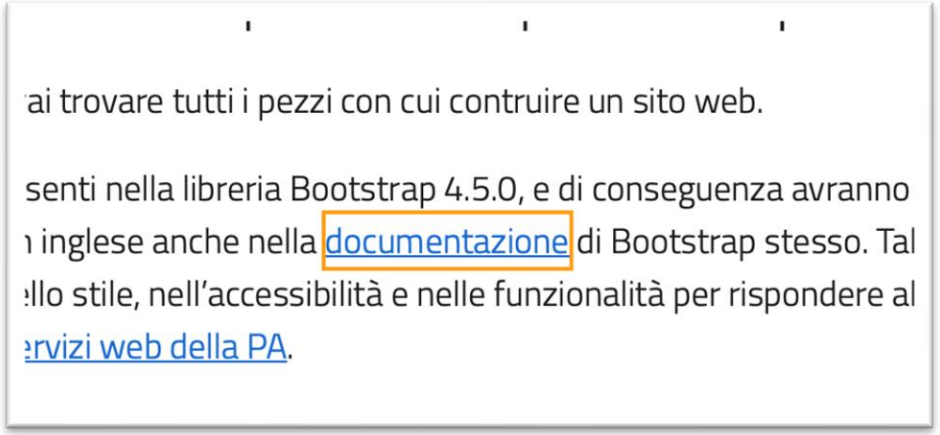


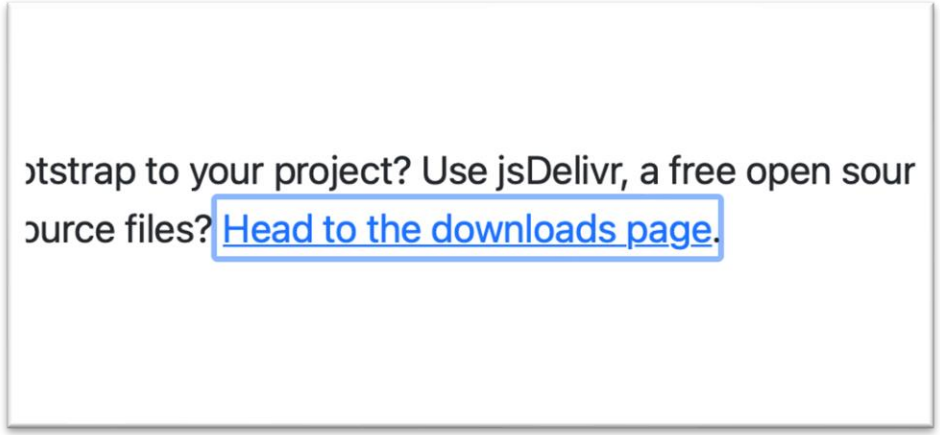
- Focus da tastiera

Per tutti gli elementi che possono ricevere il focus da tastiera (ancore, pulsanti, input, select, ecc) l'indicatore deve essere visibile, indicato spesso con la pseudo-classe CSS ***:focus {outline: 3px dashed;}** o similare.

Anche questo Criterio di Successo delle WCAG 2.1 è uno dei più noti in accessibilità, eppure anche uno dei più violati.

By default i framework hanno generalmente una corretta impostazione CSS per l'indicatore del focus.





- Focus da tastiera: problemi

I problemi legati all'indicatore del focus possono essere causati da alcuni motivi principali:

1. **Volontario azzeramento** della pseudo-classe :focus dal CSS dei framework tramite sovrascrittura (es. *:focus {outline: none;}). Erroneamente il focus visibile viene considerato un elemento non necessario ed esteticamente brutto.
2. **Non implementazione della pseudo-classe** *:focus nei sistemi CSS che non si basano su framework.
3. **Problemi legati ai comportamenti e/o modifiche dalla DOM** dovuti all'utilizzo di Framework di sviluppo quali React o similari.
   Vedere ad esempio i problemi descritti nella guida in accessibilità di React in relazione alla gestione programmatica del focus https://reactjs.org/docs/accessibility.html#programmatically-managing-focus

- Etichette: label dei campi di input

Gli elementi **label devono sempre essere presenti** nei campi di input e correttamente referenziati.
Generalmente i framework includono by default le label nelle loro specifiche.

```
<div class="form-group">
  <input type="text" class="form-control" id="exampleInputText">
  <label for="exampleInputText">Campo di tipo testuale</label>
</div>
```

```
<div class="mb-3">
  <label for="exampleFormControlInput1" class="form-label">Email address</label>
  <input type="email" class="form-control" id="exampleFormControlInput1" placeholder="nam
</div>
```

Tuttavia i problemi legati alle label sono molto comuni e per numerose tipologie

- Label: mancata referenziazione

L’elemento label deve essere correttamente referenziato al corrispondente campo di input.

```
<label for="exampleID">Campo di tipo testuale</label>
<input type="text" name=»exampleName" id="exampleID">
```

Problemi riscontrati:

1. Attributo for mancante.
2. Attributo for referenziato a una ID inesistente.
3. Attributo ID referenziato a una ID di un campo input diverso da quello corretto (problema tipico del copia e incolla del codice.
4. Attributo ID referenziato all’attributo name anziché id.

- Label: utilizzo di aria-label

Talvolta l'elemento label viene sostituito dall'attributo aria-label presente direttamente all'interno del campo di input.

```
<input type="text" name=»exampleName" aria-label="example">
```

Pur non essendo una tecnica errata talvolta può non essere accessibile perché:

1. **Riduce lo spazio selezionabile** con il mouse se applicato a un campo input di tipo checkbox o radio.
2. È una alternativa valida per le persone che utilizzano un lettore di schermo, ma **può ridurre la comprensione del contenuto** per altre persone (ad esempio disabilità cognitive).

- Label e placeholder

L’attributo placeholder non è un’alternativa dell’elemento label.

```
<input type="text" name=»exampleName" placeholder="example">
```

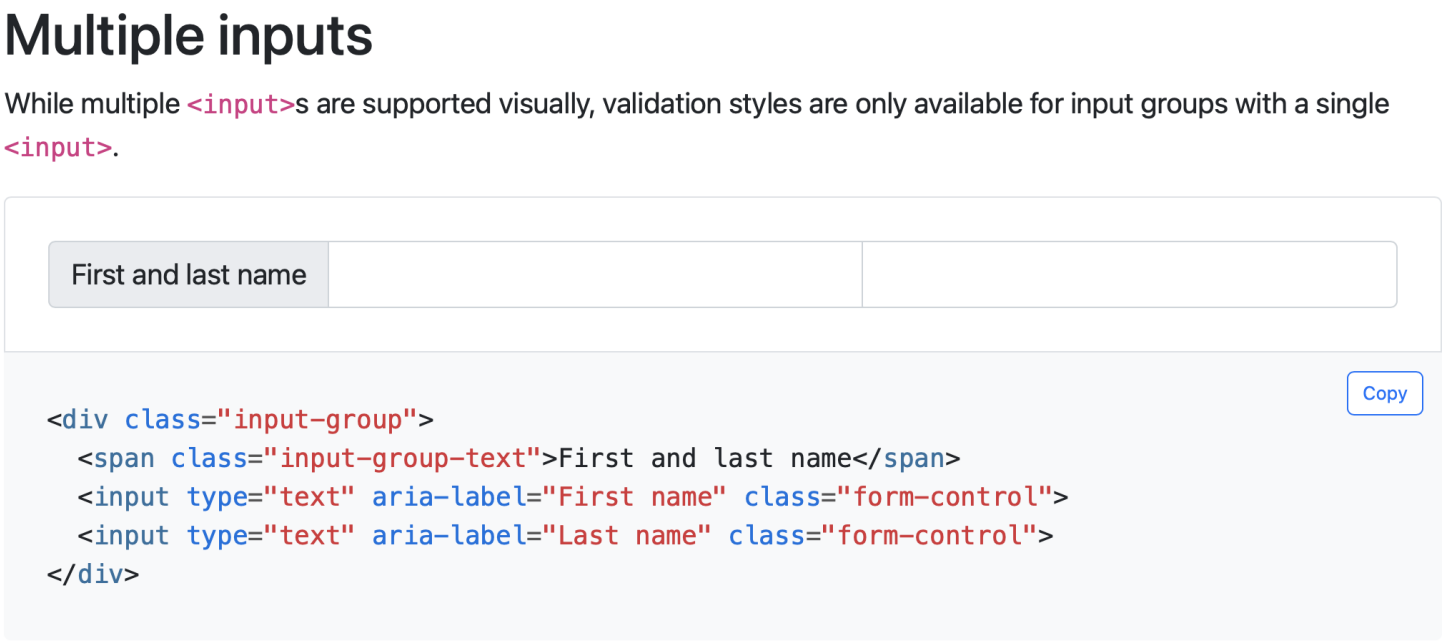
Ci sono diverse motivazioni:

1. C’è una **differenza semantica**: l’uno è un attributo, l’altro un elemento. Pertanto le tecnologie assistive interpretano i due oggetti in maniera diversa e questo può confondere gli utenti.
2. Il placeholder viene **visivamente nascosto quando l’utente inizia la digitazione** del campo. Questo impedisce agli utenti di avere sempre visibile l’etichetta del campo.
3. I placeholder sono visualizzati dai browser con **colori molto tenui** che possono impedirne la visualizzazione alle persone ipovedenti.

- Etichette: errori indotti dai framework

In **Bootstrap** esiste l'esempio **`"Multiple inputs"`** che indica due campi input etichettati attraverso l'attributo aria-label e presentati visivamente con una etichetta unica descritta con l'elemento span senza correlazione diretta.

Questa presentazione introduce un problema di accessibilità perché rende ambigua l'attribuzione visiva dell'etichetta al giusto campo. Ad esempio gli utenti possono facilmente invertire il nome e il cognome.



- Etichette: errori indotti dai framework

In **Bootstrap Italia** l'esempio Autocompete grande, è mostrato senza elemento label e senza attributo aria-label.

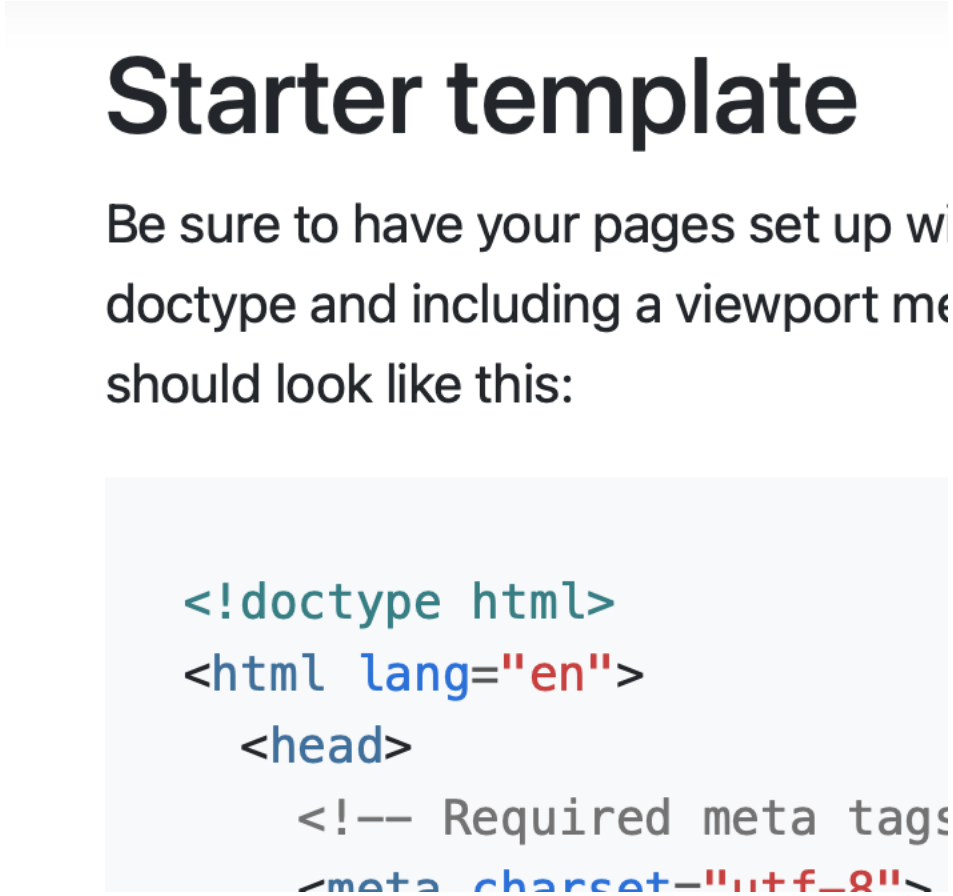
In realtà l'esempio è solo uno dei tanti dello stesso tipo e la modalità accessibile di implementazione è descritta a inizio paragrafo.

Comunque copiando acriticamente la porzione di codice si può introdurre l'errore di accessibilità.

- Lingua della pagina

L’elemento HTML del codice di una pagina deve contenere l’attributo lang.
L’attributo **deve essere valorizzato con la lingua corrente della pagina**.
Gli errori sono legati a:

1. Copia e incolla degli esempi presenti nei framework.
2. Mancata modifica del valore dell’attributo quando viene selezionato il cambio di lingua dell’applicativo.



- Il ruolo degli elementi

In HTML gli elementi possono avere una **semantica nativa**, ad esempio <H1> esprime un titolo, <NAV> un'area di navigazione, e così via.

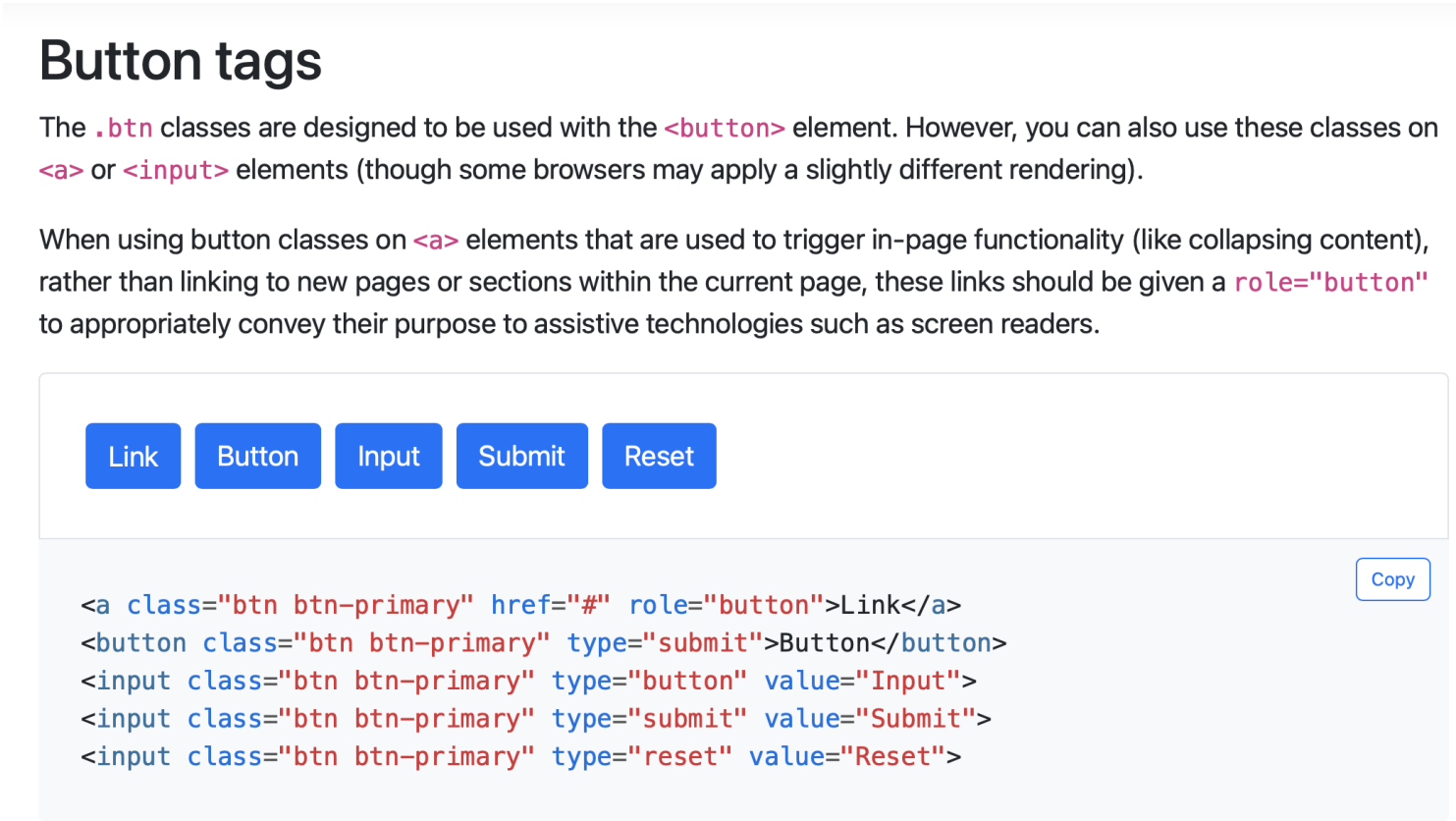
Inoltre attraverso i ruoli definiti negli attributi WAI-ARIA gli **elementi possono assumere ruoli diversi e nuovi**. Ad esempio un elemento DIV di ruolo Navigation ha lo stesso significato dell'elemento nativo NAV.

Tuttavia con i framework e i fogli di stile si possono generare errori di accessibilità.

- Il ruolo degli elementi e l’aspetto visivo

In Bootstrap con la classe btn ogni elemento può assumere le sembianze di un pulsante e questo errore semantico molto importante in accessibilità.

È frequente trovare pulsanti, esempio gli hamburger menu, con questo errore. Ciò rende invisibile il pulsante alle tecnologie assistive e non utilizzabile con la tastiera.



- Conclusione (1 di 2)

Molti requisiti di accessibilità sono molto facili da comprendere, tuttavia in fase di implementazione è possibile incorrere in errori.

Altri requisiti sono invece più complessi e talvolta è necessario ricorre agli esperti di accessibilità per garantire un corretto sviluppo accessibile.

- Conclusione (2 di 2)

Sviluppare accessibile non è un’alternativa migliore,
ma è l’unico modo di sviluppare bene.

www.agid.gov.it

Fabrizio Caccavello
fabrizio.caccavello@agid.gov.it