

REST vs SOAP: Integrazione API in PHP

Analisi comparativa tramite esempi pratici



Obiettivo della presentazione

Mostrare come interagire con API REST e SOAP in PHP usando cURL con esempi commentati



Tecnologie usate

Utilizzo di cURL per HTTP request, json_decode per REST e XML+regex per SOAP



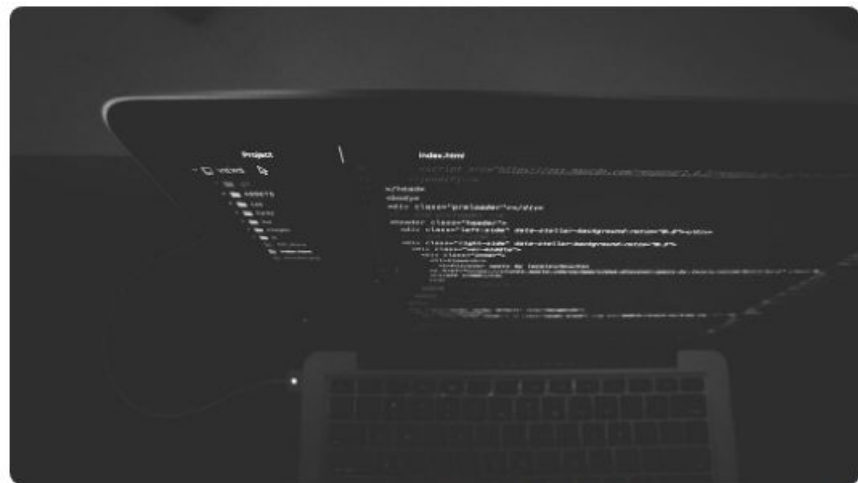
Confronto diretto

Analisi pratica delle differenze tra REST e SOAP in termini di implementazione e utilizzo

Introduzione alla Comunicazione API in PHP

REST e SOAP con cURL

- **Obiettivo della presentazione:** Comprendere come inviare richieste REST e SOAP in PHP usando la libreria cURL per accedere a dati remoti.
- **Importanza della comunicazione API:** Le API sono fondamentali per integrare servizi web, scambiare dati e creare applicazioni interconnesse.
- **Struttura della lezione:** Analisi dettagliata di due script PHP: uno per REST (JSON) e uno per SOAP (XML), con spiegazioni riga per riga.



REST API in PHP: URL, Inizializzazione e Configurazione

client_rest.php - Parte 1



Definizione URL API

L'endpoint

`https://jsonplaceholder.typicode.com/users/` restituisce i dati in formato JSON dell'utente con ID5.



Inizializzazione cURL

La funzione ``curl_init()`` apre una nuova sessione HTTP, come avviare una telefonata.



Impostazione URL

``curl_setopt(..., CURLOPT_URL, $url)`` assegna l'URL alla sessione cURL per definire il destinatario della richiesta.

REST API in PHP: Ricezione e Gestione della Risposta

client_rest.php - Parte 2



CURLOPT_RETURNTRANSFER

Imposta la risposta come stringa con ``curl_setopt(..., CURLOPT_RETURNTRANSFER, true)`` per elaborarla in seguito.



Esecuzione richiesta

``curl_exec($ch)`` invia la richiesta all'URL specificato e memorizza la risposta JSON in una variabile.



Gestione errori cURL

``curl_errno()`` e ``curl_error()`` permettono di identificare e stampare errori di connessione HTTP.

REST API in PHP: Chiusura, Decodifica e Output

client_rest.php - Parte 3

- **Chiusura sessione cURL:** ``curl_close($ch)`` chiude la connessione HTTP e libera le risorse di sistema.
- **Decodifica JSON:** ``json_decode($response, true)`` converte la risposta JSON in un array PHP associativo.
- **Visualizzazione dati:** Estrazione e stampa di name, email e city da ``$data``, per mostrare le informazioni utente.



Riepilogo: Comunicazione REST in PHP

Vantaggi e Buone Pratiche



Chiarezza del flusso

Ogni fase – apertura, configurazione, esecuzione, chiusura – è gestita in modo esplicito e tracciabile.



Controllo completo

cURL offre piena flessibilità nella gestione di HTTP, header, metodi, payload e parsing della risposta.



Manutenibilità e debugging

Gestione errori integrata, output testuale, e struttura modulare migliorano la capacità di individuare problemi.

SOAP in PHP: URL e Richiesta XML

client_soap.php - Parte 1

- **Definizione URL SOAP:** Il servizio ``http://www.w3schools.com/xml/tempconvert.aspx`` consente conversioni di temperatura via SOAP.
- **Struttura XML SOAP:** Creazione del messaggio XML per la funzione ``CelsiusToFahrenheit``, secondo il protocollo SOAP.
- **Inizializzazione cURL:** ``curl_init()`` apre la sessione HTTP per la richiesta SOAP, come nel caso REST.

```
1 // Promise from setTimeout
2 const afterSomeTime = (time) => new Promise(resolve => {
3   setTimeout(() => {
4     resolve(true);
5   }, time);
6 });
7 const callAfterSomeTime = (callback, time) => afterSomeTime(time).then(callback);
8
9 callAfterSomeTime(() => console.log('Hello after 1500ms'), 1500);
10
11 const getData = async (url) => fetch(url);
12
13 document
14   .querySelector('#submit')
15   .addEventListener('click', function() {
16     const name = document.querySelector("#name").value;
17
18     // send to backend
19     const user = await fetch('/users?name=' + name);
20     const post = await fetch('/posts?userId=' + user.id);
21     const posts = post.json();
22     console.log(posts[0].id);
23   });
```

SOAP in PHP: POST, Header e XML

client_soap.php - Parte 2



Impostazione metodo POST

``CURLOPT_POST`` forza l'uso del metodo HTTP POST, essenziale per il protocollo SOAP.



Header SOAP

Header richiesti: ``Content-Type: text/xml``, ``SOAPAction``, e ``Content-Length`` basato sull'XML.



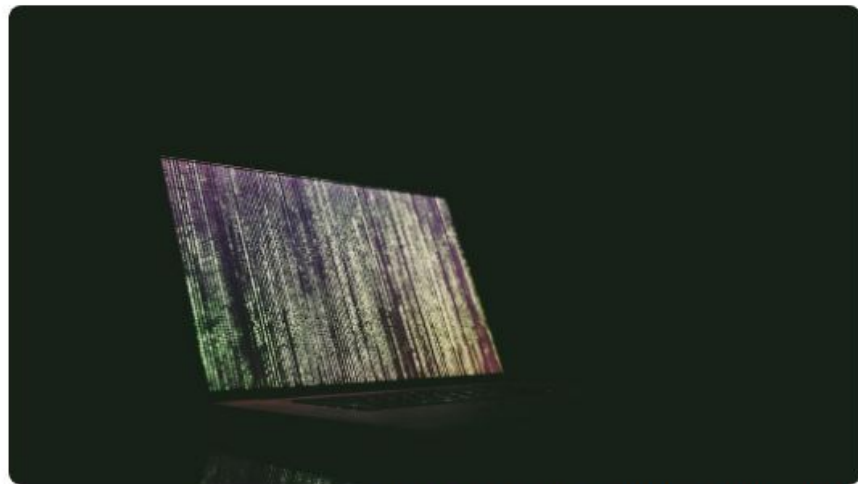
Corpo della richiesta

Con ``CURLOPT_POSTFIELDS``, il corpo XML viene incluso nella richiesta HTTP.

SOAP in PHP: Esecuzione, Errori e Parsing

client_soap.php - Parte 3

- **Esecuzione richiesta:** ``curl_exec($ch)`` invia il messaggio XML e riceve la risposta SOAP in formato stringa.
- **Gestione errori cURL:** Controllo con ``curl_errno()`` e ``curl_error()`` per diagnosticare eventuali problemi di comunicazione.
- **Estrazione risultato:** Uso di ``preg_match()`` per recuperare il valore Fahrenheit da `` nell'XML.



Riepilogo: Integrazione SOAP in PHP

Punti Chiave e Vantaggi



Struttura rigorosa

SOAP richiede formattazione XML ben definita, rendendolo ideale per contesti enterprise o normati.



Controllo su headers e metodi

Gestione dettagliata di header HTTP e azioni remote tramite ``SOAPAction``.



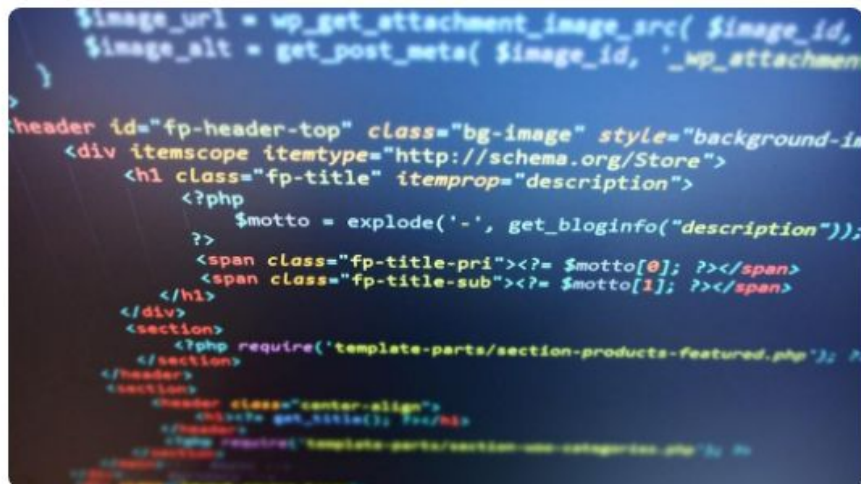
Parsing mirato

Utilizzo di ``preg_match`` per estrarre dati specifici da un XML ben strutturato.

Conclusione Generale: REST e SOAP in PHP

Riflessioni Finali e Best Practice

- **Conoscenza solida dei protocolli:** REST e SOAP sono strumenti potenti per integrare applicazioni: REST per leggerezza, SOAP per rigore.
- **cURL come base operativa:** Gestione di richieste HTTP in modo personalizzato: headers, metodi, payload e parsing risposta.



```
$image_url = wp_get_attachment_image_src( $image_id,
$image_alt = get_post_meta( $image_id, '_wp_attachment
)
}
header id="fp-header-top" class="bg-image" style="background-image:
<div itemscope itemtype="http://schema.org/Store">
  <h1 class="fp-title" itemprop="description">
    <?php
      $motto = explode('-', get_bloginfo("description"));
    >
    <span class="fp-title-pri"><? $motto[0]; ?></span>
    <span class="fp-title-sub"><? $motto[1]; ?></span>
  </h1>
  <section>
    <?php require('template-parts/section-products-featured.php'); ?>
  </section>
</header>
<section>
  <header class="center-align">
    <h2><? get_bloginfo('title'); ?></h2>
  </header>
  <?php require('template-parts/section-products-categories.php'); ?>
</section>
</div>
```