Web APIs: Complete Guide

What are Web APIs?

API = Application Programming Interface

A **Web API** is a service that allows different software applications to communicate with each other over the internet using HTTP/HTTPS protocols.

Think of it like a **waiter in a restaurant**:

- You (your app) tell the waiter (API) what you want
- The waiter takes your request to the kitchen (server)
- The kitchen prepares your order (processes data)
- The waiter brings back your food (returns response)

Simple Example

javascript

```javascript
// You ask the API for weather data
fetch('https://api.weather.com/current?city=Phoenix')
  .then(response => response.json())
  .then(data => {
    console.log(data.temperature); // API gives you the answer
  });
```

How Web APIs Work

The Request-Response Cycle

javascript

```javascript
// 1. Your app makes a REQUEST
fetch('https://api.example.com/users/123', {
  method: 'GET',
  headers: {
```

```
    'Content-Type': 'application/json',
    'Authorization': 'Bearer YOUR_API_KEY'
  }
})

// 2. Server processes the request

// 3. Server sends back a RESPONSE (usually in JSON)
.then(response => response.json())
.then(data => {
  console.log(data);
  // {
  //   "id": 123,
  //   "name": "Alice",
  //   "email": "alice@example.com"
  // }
});
```

Common HTTP Methods

javascript

JavaScript

```
// GET - Retrieve data (read)
fetch('https://api.example.com/products')

// POST - Create new data
fetch('https://api.example.com/products', {
  method: 'POST',
  body: JSON.stringify({ name: "New Product", price: 29.99 })
})

// PUT - Update existing data (replace)
fetch('https://api.example.com/products/123', {
  method: 'PUT',
  body: JSON.stringify({ name: "Updated Product", price: 39.99 })
```

```
})

// PATCH - Partially update data
fetch('https://api.example.com/products/123', {
  method: 'PATCH',
  body: JSON.stringify({ price: 39.99 })
})

// DELETE - Remove data
fetch('https://api.example.com/products/123', {
  method: 'DELETE'
})
```

Why JSON is Used in APIs

Advantages of JSON Format

1. Human-Readable and Easy to Understand

json

```JSON
{
  "user": {
    "name": "John",
    "age": 30,
    "email": "john@example.com"
  }
}
```

2. Language-Independent Works with JavaScript, Python, Java, PHP, Ruby, C#, etc.

3. Lightweight Less data to transfer compared to XML

4. Direct Mapping to JavaScript Objects

javascript

```javascript
const response = '{"name":"John","age":30}';
const user = JSON.parse(response); // Instant object!
console.log(user.name); // "John"
```

JSON vs XML Comparison

javascript

```javascript
// JSON - Compact and clean
{
  "product": {
    "name": "Laptop",
    "price": 999.99,
    "inStock": true
  }
}

// XML - More verbose
<product>
  <name>Laptop</name>
  <price>999.99</price>
  <inStock>true</inStock>
</product>
```

JSON is typically 30-50% smaller than equivalent XML!

Advantages of Using Web APIs

1. Don't Reinvent the Wheel

Instead of building everything yourself, use existing services:

javascript

```javascript
// Instead of building your own payment system...
```

```
// Use Stripe API
const payment = await stripe.charges.create({
  amount: 2000,
  currency: 'usd',
  source: 'tok_visa'
});

// Instead of building your own map...
// Use Google Maps API
const map = new google.maps.Map(document.getElementById('map'), {
  center: { lat: 33.4484, lng: -112.0740 }, // Phoenix
  zoom: 10
});
```

2. Real-Time Data Access

Get up-to-date information instantly:

javascript

JavaScript

```
// Stock prices
fetch('https://api.example.com/stocks/AAPL')
  .then(res => res.json())
  .then(data => console.log(`Apple stock: $${data.price}`));

// Weather updates
fetch('https://api.weather.com/current')
  .then(res => res.json())
  .then(data => console.log(`Temperature: ${data.temp}°F`));
```

3. Separation of Concerns

Frontend and backend can work independently:

javascript

```
JavaScript
// Frontend developers work here
const app = {
  async loadUsers() {
    const users = await fetch('/api/users').then(r => r.json());
    this.displayUsers(users);
  }
};

// Backend developers work on the API separately
// They can change database, logic, etc. without breaking frontend
```

```
### 4. **Scalability**
One API can serve multiple applications:
```

```
Same API →  Web App
         →  Mobile App (iOS)
         →  Mobile App (Android)
         →  Desktop App
         →  Third-party integrations
```

5. Security

Control who accesses your data:

javascript

```
JavaScript
// API requires authentication
fetch('https://api.example.com/private-data', {
  headers: {
    'Authorization': 'Bearer YOUR_SECRET_TOKEN'
  }
})
```

Cost of Web APIs

Free Tier APIs

Many APIs offer free usage with limits:

javascript

```javascript
// Example: OpenWeatherMap
// Free tier: 1,000 calls/day
const apiKey = 'YOUR_FREE_API_KEY';
fetch(`https://api.openweathermap.org/data/2.5/weather?q=Phoenix&
appid=${apiKey}`)
```

Popular Free/Freemium APIs:

- OpenWeatherMap (weather data)
- JSONPlaceholder (testing/prototyping)
- NASA APIs (space data)
- GitHub API (repository data)
- News API (news articles)

Paid APIs

For production/heavy usage:

API	Free Tier	Paid Plans
Google Maps	$200/month credit	$7 per 1,000 requests
Stripe (Payments)	Free to integrate	2.9% + 30¢ per transaction
SendGrid (Email)	100 emails/day	$15/month for 40,000 emails
Twilio (SMS)	Trial credit	$0.0075 per SMS
AWS	Free tier 12 months	Pay as you go

Cost Considerations

javascript

JavaScript

```
// Example: Planning API costs
const monthlyUsers = 10000;
const apiCallsPerUser = 50;
const totalCalls = monthlyUsers * apiCallsPerUser; // 500,000
calls

// If API costs $5 per 10,000 calls
const monthlyCost = (totalCalls / 10000) * 5; // $250/month
```

Real-World Use Case Examples

1. E-Commerce Application

javascript

JavaScript

```
// Product catalog from your backend API
async function loadProducts() {
  const response = await
fetch('https://mystore.com/api/products');
  const products = await response.json();

  // Response JSON format:
  // {
  //   "products": [
  //     {
  //       "id": 1,
  //       "name": "Laptop",
  //       "price": 999.99,
  //       "category": "Electronics",
  //       "inStock": true,
  //       "image": "https://..."
  //     }
  //   ],
  //   "total": 150,
  //   "page": 1
```

```
  // }

  displayProducts(products.products);
}

// Process payment via Stripe API
async function processPayment(amount, token) {
  const response = await
fetch('https://api.stripe.com/v1/charges', {
    method: 'POST',
    headers: {
      'Authorization': 'Bearer sk_test_...',
      'Content-Type': 'application/json'
    },
    body: JSON.stringify({
      amount: amount * 100, // Convert to cents
      currency: 'usd',
      source: token
    })
  });

  const result = await response.json();
  // {
  //   "id": "ch_abc123",
  //   "status": "succeeded",
  //   "amount": 99900
  // }

  return result;
}
```

2. Weather Dashboard

javascript

JavaScript

```
async function getWeather(city) {
  const apiKey = 'your_api_key';
  const url =
`https://api.openweathermap.org/data/2.5/weather?q=${city}&appid=
${apiKey}&units=imperial`;

  const response = await fetch(url);
  const data = await response.json();

  // JSON Response format:
  // {
  //   "name": "Phoenix",
  //   "main": {
  //     "temp": 95.5,
  //     "feels_like": 98.2,
  //     "humidity": 20
  //   },
  //   "weather": [
  //     {
  //       "main": "Clear",
  //       "description": "clear sky",
  //       "icon": "01d"
  //     }
  //   ],
  //   "wind": {
  //     "speed": 8.5
  //   }
  // }

  return {
    city: data.name,
    temperature: data.main.temp,
    description: data.weather[0].description,
    humidity: data.main.humidity
  };
}
```

```
// Usage
getWeather('Phoenix').then(weather => {
  console.log(`${weather.city}: ${weather.temperature}°F,
${weather.description}`);
});
```

3. Social Media Integration

javascript

JavaScript

```
// Post to Twitter via API
async function postTweet(message) {
  const response = await
fetch('https://api.twitter.com/2/tweets', {
    method: 'POST',
    headers: {
      'Authorization': 'Bearer YOUR_ACCESS_TOKEN',
      'Content-Type': 'application/json'
    },
    body: JSON.stringify({
      text: message
    })
  });

  const result = await response.json();
  // {
  //   "data": {
  //     "id": "1234567890",
  //     "text": "Hello from my app!"
  //   }
  // }

  return result.data;
}
```

4. Restaurant App with Multiple APIs

javascript

```javascript
async function buildRestaurantApp(location) {
  // 1. Get restaurants nearby (Yelp API)
  const restaurants = await
fetch(`https://api.yelp.com/v3/businesses/search?location=${locat
ion}`, {
    headers: { 'Authorization': 'Bearer YOUR_YELP_KEY' }
  }).then(r => r.json());

  // JSON: { "businesses": [{ "name": "...", "rating": 4.5, ... }]
}

  // 2. Show on map (Google Maps API)
  const map = new
google.maps.Map(document.getElementById('map'));
  restaurants.businesses.forEach(restaurant => {
    new google.maps.Marker({
      position: { lat: restaurant.coordinates.latitude, lng:
restaurant.coordinates.longitude },
      map: map
    });
  });

  // 3. Allow reservations (OpenTable API)
  const makeReservation = async (restaurantId, time, guests) => {
    return fetch('https://api.opentable.com/reservations', {
      method: 'POST',
      body: JSON.stringify({ restaurantId, time, guests })
    }).then(r => r.json());
  };

  // 4. Send confirmation (Twilio SMS API)
  const sendConfirmation = async (phoneNumber, message) => {
```

```
    return
fetch('https://api.twilio.com/2010-04-01/Accounts/YOUR_ACCOUNT/Me
ssages.json', {
      method: 'POST',
      body: JSON.stringify({
        To: phoneNumber,
        From: '+1234567890',
        Body: message
      })
    });
  };
}
```

5. Real-Time Collaboration Tool

javascript

JavaScript

```
// Using Firebase Realtime Database API
const database = firebase.database();

// Write data (JSON format)
function saveDocument(docId, content) {
  database.ref('documents/' + docId).set({
    content: content,
    lastModified: Date.now(),
    author: currentUser.id
  });

  // Stored in Firebase as JSON:
  // {
  //   "documents": {
  //     "doc123": {
  //       "content": "Hello world",
  //       "lastModified": 1704153600000,
  //       "author": "user456"
```

```
  //     }
  //   }
  // }
}

// Listen for real-time updates
database.ref('documents/' + docId).on('value', (snapshot) => {
  const data = snapshot.val();
  updateEditor(data.content);
});
```

6. Fitness Tracker App

javascript

```
JavaScript
// Track workout using multiple APIs
async function logWorkout(workout) {
  // 1. Save to your database API
  const saved = await fetch('https://myapp.com/api/workouts', {
    method: 'POST',
    body: JSON.stringify({
      type: workout.type,
      duration: workout.duration,
      calories: workout.calories,
      date: new Date().toISOString()
    })
  }).then(r => r.json());

  // 2. Sync with fitness device (Fitbit API)
  await fetch('https://api.fitbit.com/1/user/-/activities.json',
{
    method: 'POST',
    headers: { 'Authorization': 'Bearer FITBIT_TOKEN' },
    body: JSON.stringify({
      activityId: 90009, // Running
```

```
      durationMillis: workout.duration * 60000
    })
  });

  // 3. Share achievement (Facebook API)
  if (workout.calories > 500) {
    await fetch('https://graph.facebook.com/me/feed', {
      method: 'POST',
      body: JSON.stringify({
        message: `Just burned ${workout.calories} calories! 🔥`
      })
    });
  }

  return saved;
}
```

API Response Format (JSON Structure)

Standard Success Response

json

```JSON
{
  "status": "success",
  "data": {
    "id": 123,
    "name": "Product Name",
    "price": 29.99
  },
  "message": "Product retrieved successfully",
  "timestamp": "2024-02-02T10:30:00Z"
}
```

Standard Error Response

json

```json
{
  "status": "error",
  "error": {
    "code": 404,
    "message": "Product not found",
    "details": "No product exists with ID 123"
  },
  "timestamp": "2024-02-02T10:30:00Z"
}
```

Paginated Response

json

```json
{
  "data": [
    { "id": 1, "name": "Item 1" },
    { "id": 2, "name": "Item 2" }
  ],
  "pagination": {
    "page": 1,
    "perPage": 20,
    "total": 150,
    "totalPages": 8
  }
}
```

Best Practices

1. Always Handle Errors

javascript

```javascript
async function fetchUserData(userId) {
```

```
  try {
    const response = await
fetch(`https://api.example.com/users/${userId}`);

    if (!response.ok) {
      throw new Error(`HTTP error! status: ${response.status}`);
    }

    const data = await response.json();
    return data;
  } catch (error) {
    console.error('Failed to fetch user:', error);
    return null;
  }
}
```

2. Use Environment Variables for API Keys

javascript

```
JavaScript
// Don't hardcode API keys!
// ❌ Bad
const apiKey = 'sk_live_abc123xyz';

// ✅ Good
const apiKey = process.env.API_KEY;
```

3. Implement Rate Limiting Awareness

javascript

```
JavaScript
// Respect API rate limits
const delay = ms => new Promise(resolve => setTimeout(resolve,
ms));
```

```
async function fetchMultipleUsers(userIds) {
  const results = [];

  for (const id of userIds) {
    const user = await
fetch(`https://api.example.com/users/${id}`);
    results.push(await user.json());
    await delay(100); // Wait 100ms between requests
  }

  return results;
}
```

Summary

Web APIs + JSON = Modern Web Development

- **APIs** let different applications talk to each other
- **JSON** is the universal language they use
- **Benefits**: Save time, access real-time data, scale easily
- **Cost**: Ranges from free to pay-per-use
- **Use Cases**: Everything from payments to weather to social media

This combination powers virtually every modern web and mobile application you use daily!