Response Object

Sending response back to Client

In an Express.js application, you send a response back to the client using the `res` (response) object. The `res` object provides various methods to send different types of responses, such as text, JSON, HTML, and more. Here are some common ways to send responses back to the client:

1. **Sending Plain Text:**
    - Use the `res.send` method to send plain text.

    ```
    javascriptCopy code
    app.get('/', (req, res) => {
      res.send('Hello, this is a plain text response!');
    });
    ```

2. **Sending JSON:**
    - Use the `res.json` method to send a JSON response.

    ```
    javascriptCopy code
    app.get('/api/data', (req, res) => {
      const data = { message: 'This is a JSON response.' };
      res.json(data);
    });
    ```

3. **Sending HTML:**
    - Use the `res.send` method to send HTML content.

    ```
    javascriptCopy code
    app.get('/html', (req, res) => {
      const htmlContent = '<h1>This is an HTML response</h1>';
      res.send(htmlContent);
    });
    ```

4. **Redirecting:**
    - Use the `res.redirect` method to redirect the client to a different URL.

    ```
    javascriptCopy code
    app.get('/redirect', (req, res) => {
      res.redirect('/new-location');
    });
    ```

5. **Sending Status Codes:**
    - Use the `res.status` method to set the HTTP status code.

    ```
    javascriptCopy code
    app.get('/not-found', (req, res) => {
      res.status(404).send('Page not found');
    });
    ```

6. **Sending Files:**
    - Use the `res.sendFile` method to send files.

```
javascriptCopy code
const path = require('path');

app.get('/file', (req, res) => {
  const filePath = path.join(__dirname, 'files', 'example.txt');
  res.sendFile(filePath);
});
```

7. **Setting Headers:**
   - Use the `res.set` method to set HTTP headers.

```
javascriptCopy code
app.get('/custom-header', (req, res) => {
  res.set('X-Custom-Header', 'Custom Header Value');
  res.send('Response with a custom header');
});
```

These examples showcase various ways to send responses back to the client based on different scenarios. The `res` object provides a versatile set of methods to handle a wide range of response types. Depending on the use case, you can choose the appropriate method to send the desired response to the client.