Automatic JSON Conversion of Lists in Spring Boot

Key Components and Mechanism

Spring Boot's automatic JSON conversion relies on the following:

- **@RestController** and **@ResponseBody** Annotations:
  - @RestController combines @Controller and @ResponseBody, indicating that method return values should be written directly to the HTTP response body.
  - @ResponseBody can also be used on individual controller methods within a @Controller.
- **Default Message Converters:**
  - Spring Boot includes default message converters to handle data format conversions.
  - MappingJackson2HttpMessageConverter uses the Jackson library for JSON serialization/deserialization.

Process Flow

1. **Controller Method Execution:** A controller method returns a List of Java objects.
2. **Message Converter Selection:** Spring selects MappingJackson2HttpMessageConverter based on the Accept header and return type.
3. **JSON Serialization:** Jackson converts the List into a JSON array:
   - Iterates through the List.
   - Serializes each object's properties into a JSON object.
4. **Response Generation:** The JSON string is written to the HTTP response body with the Content-Type set to application/json.

Conclusion

Spring Boot's automatic JSON conversion, facilitated by @ResponseBody and MappingJackson2HttpMessageConverter, streamlines RESTful API development. Developers can focus on business logic, as the framework handles JSON serialization.