Support for auto-validation in Swagger annotations.

In 7.2, we're introducing auto-validation of Swagger annotations. What this means is that the various Swagger annotations are not just used for populating the Swagger JSON / UI, but are actively used in serializing/parsing/validating the HTTP parts.

For example:

```java
@RestMethod(
    name = "GET",
    path = "/pet/findByStatus",
    summary = "Finds Pets by status",
    description = "Multiple status values can be provided with comma separated strings.",
    @MethodSwagger(
        tags = "pet",
        value = {
            "security": [{petstore_auth: ['write:pets', 'read:pets']}]}
    )
)
public Collection<Pet> findPetsByStatus(
    @Query(
        name = "status",
        description = "Status values that need to be considered for filter.",
        required = true,
        type = "array",
        collectionFormat = "csv",
        items = @Items(
            type = "string",
            _enum = "AVAILABLE,PENDING,SOLD",
            _default = "AVAILABLE"
        ),
        example = "AVAILABLE,PENDING"
    )
    PetStatus[] status
) throws NotAcceptable {
    return store.getPetsByStatus(status);
}
```

This example shows a parameter of type `PetStatus[]` (which happens to be enums, but could also be POJOs as well). The `type` and `collectionFormat` attributes tell the framework to parse the incoming value as a comma-delimited list. The `_enum` shows the possible valid values. A `BadRequest` is thrown if any values don't match.

Part of this change includes combining the client-side and server-side annotations into a single set of annotations (no more confusing `org.apache.juneau.remoteable.Query` with `org.apache.juneau.rest.annotation.Query`).

Features include:

- Support for all of Swagger 2.0.
- Auto-validation for:
  - Numeric values (`minimum`, `maximum`, `exclusiveMinimum`, `exclusiveMaximum`, `multipleOf`).
  - String values (`minLength`, `maxLength`, `enum`).
  - Collection values (`items`, `minItems`, `maxItems`, `uniqueItems`).
  - Object values (`properties`, `maxProperties`, `minProperties`, `additionalProperties`).
  - Existence of values (`required`, `allowEmptyValue`).
- Support for both client-side and server-side annotations.
- Works on the `Body` annotation as well. If the media-type of the body does not match an existing serializer/parser, then the Swagger rules are used for marshalling.
- UON notation is still supported, so you can still represent arbitrary POJOs as any of the HTTP parts. However, you have to explicitly specify `form at="uon"`. 