Designing Associations between Resources

Introduction

Defining Relationships between Resources

Defining Relationships between Resources

Find relationship between resources

Use URL navigation for sub-objects
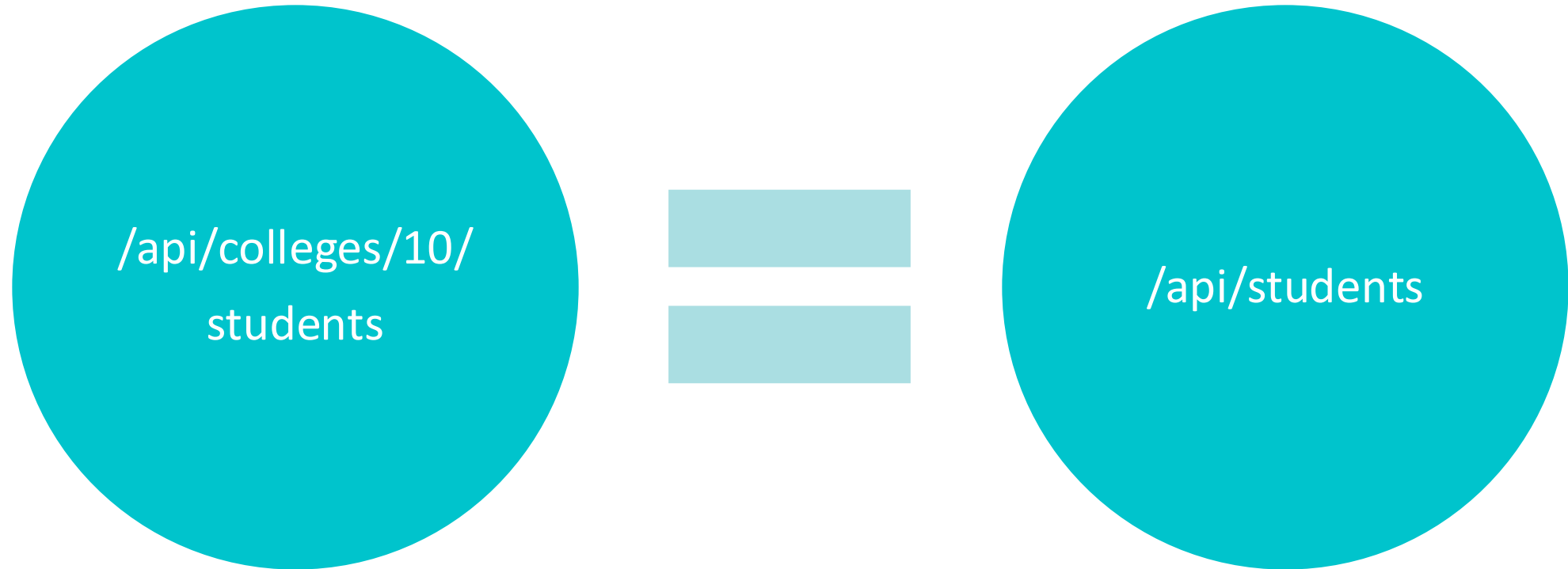
Use URL Navigation for Sub-objects

/api/colleges/10

College object whose ID is 10

/api/colleges/10/students

Students of the college whose ID is 10

Sub-objects should return same format as collections

Format of Sub-objects

/api/colleges/10/students = /api/students

Defining Relationships between Resources

Find relationship between resources

Use URL navigation for sub-objects

Multiple associations at the same level

Multiple Associations at Same Level

/api/colleges/10	College object whose ID is 10
/api/colleges/10/students	Students of the college whose ID is 10
/api/colleges/10/staff	Staffs of the college whose ID is 10

STEP 8: Identify the Association for Each Resource

URL Shouldn't be Complex

URLs shouldn't be more complex than
collection/item/collection

URL Complexity

Good URL Design

/api/colleges
/api/colleges/10
/api/colleges/10/students

URL Complexity

Good URL Design

/api/colleges
/api/colleges/10
/api/colleges/10/students

Bad URL Design

/api/colleges/10/students/5
/api/colleges/10/students/5/courses
/api/colleges/10/students/5/hobbies

Redesign Complex URLs

Bad URL Design

- /api/colleges/10/students/5/courses
- /api/colleges/10/students/5/hobbies

Redesign Complex URLs

Bad URL Design

- /api/colleges/10/students/5/courses
- /api/colleges/10/students/5/hobbies

Redesigned URLs

- /api/colleges/10/students
- /api/students/5/courses
- /api/students/5/hobbies

STEP 9: Check for the URL Complexity

Combine Related Resources

Combine Related Resources

Before:

/api/students/5/courses

/api/students/5/hobbies

Combine Related Resources

Before:

/api/students/5/courses

/api/students/5/hobbies

After:

/api/students/5

{

    Basic student details

    ...

    courses

    ...

    hobbies

    ...

}

Designing Associations between Resources

Summary