释锐数字校园开放接口v1.8：开发者调用指南

原创：释锐

> 18个已开放接口的用途、简介、输入参数、输出数据全解析，附带完整JavaScript调用示例

文档版本：v1.8
更新日期：2026年03月03日

接口调用基础说明

调用前须知

- **授权方式**：IP白名单模式，需提前将第三方系统IP告知释锐
- **模块依赖**：必须购买相应应用模块方可调用对应API
- **请求方法**：所有接口均使用POST方法
- **数据格式**：请求体支持JSON或form-urlencoded，响应为JSON格式

通用响应码说明

success/msg格式（部分接口）

- `success: true` - 操作成功，`msg`中包含data的JSON字符串
- `success: false` - 操作失败，`msg`中包含错误信息

code/msg格式（部分接口）

- `code: 200` - 成功，`msg`中包含数据列表的JSON字符串
- `code: 400` - 客户端错误（参数问题）
- `code: 501` - 服务端错误或无数据

1. 用户基本信息查询接口

接口描述：根据用户ID获取单个用户的完整基本信息，包括账号、姓名、性别、身份证件、联系方式等。

接口地址：`http://{ip}:{port}/oa/api/query/user`

输入参数详解

userId `String` **必填**

- 说明：用户的登录账号，系统唯一标识
- 示例：`"teac"`、`"zhangsan"`

输出参数详解

外层响应参数

- **success** `Boolean` **必含**：true-成功，false-失败
- **msg** `String` **必含**：成功时返回data的JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON对象）

账户基本信息

- **jctb020120** `String` **必含**：用户标识（登录账号），示例：`"teac"`
- **jctb020101** `String` **必含**：姓名，示例：`"教师甲"`
- **jctb020102** `String` **可选**：英文姓名，示例：`"John"`
- **jctb020103** `String` **可选**：姓名拼音，示例：`"jiaoshijia"`
- **jctb020104** `String` **可选**：曾用名，示例：`"张三"`
- **userType** `Integer` **必含**：用户类型（0-学生，1-教职工，2-家长）
- **password** `String` **必含**：加密密码（MD5单向加密），示例：`"13B3E9DC60323BE98717EC2B27572A87"`
- **username** `String` **可选**：合育会议账号，示例：`"huiyi001"`
- **createDate** `Timestamp` **必含**：创建时间（毫秒级时间戳），示例：`1757062642000`

身份信息

- **jctb020105** `String` **可选**：性别码（0-未知，1-男性，2-女性，9-未说明）
- **jctb020106** `String` **可选**：出生日期，示例：`"1990-01-01"`
- **jctb020107** `String` **可选**：出生地码，详见码表
- **jctb020108** `String` **可选**：籍贯，示例：`"上海"`
- **jctb020109** `String` **可选**：民族码（01-汉族，详见码表）
- **jctb020110** `String` **可选**：国籍/地区码，详见码表
- **jctb020111** `String` **可选**：身份证件类型码，详见码表
- **jctb020112** `String` **可选**：身份证件号，示例：`"310101199001011234"`
- **jctb020113** `String` **可选**：婚姻状况码，详见码表
- **jctb020114** `String` **可选**：港澳台侨外码，详见码表
- **jctb020115** `String` **可选**：政治面貌码，详见码表
- **jctb020116** `String` **可选**：健康状况码，详见码表

- **jctb020117** `String` **可选**：信仰宗教码，详见码表
- **jctb020118** `String` **可选**：血型码，详见码表
- **jctb020121** `String` **可选**：身份证件有效期，示例：`"2025-12-31"`
- **jctb020122** `String` **可选**：独生子女标志（"1"-是，"0"-否）

联系方式与地址

- **zxxs010104** `String` **可选**：住址，示例：`"上海市某某区某某路1号"`
- **zxxs010105** `String` **可选**：户口所在地，示例：`"上海市某某派出所"`
- **zxxs010106** `String` **可选**：户口性质码，详见码表
- **zxxs010108** `String` **可选**：自定义用户ID（唯一），示例：`"EMP001"`
- **zxxs010109** `String` **可选**：手机号码（唯一），示例：`"13800138000"`
- **zxxs010110** `String` **可选**：通信地址，示例：`"上海市某某区某某路1号"`
- **zxxs010111** `String` **可选**：微信openid码，示例：`"o6_bmjrPTlm6_2sgVt7hMZOPfL2M"`
- **zxxs010112** `String` **可选**：电子信箱（唯一），示例：`"teacher@school.com"`
- **zxxs010113** `String` **可选**：预留字段，批量导入时临时使用

JavaScript调用示例

```javascript
/**
 * 接口1：查询用户基本信息
 * @param {string} userId - 用户的登录账号（必填）
 * @returns {Promise<Object>} 用户基本信息对象
 */
async function queryUser(userId) {
  const url = 'http://your-server:port/oa/api/query/user';

  // 验证必填参数
  if (!userId) {
    throw new Error('userId参数是必需的');
  }

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/x-www-form-urlencoded',
      },
      body: new URLSearchParams({ userId })
    });

```

```
    const result = await response.json();

    // 解析响应
    if (result.success === true) {
      // msg中包含的是字符串化的JSON，需要解析
      const userData = JSON.parse(result.msg);
      console.log('用户信息查询成功:', userData);

      // 输出各个字段示例
      console.log('登录账号(jctb020120):', userData.jctb020120);
      console.log('姓名(jctb020101):', userData.jctb020101);
      console.log('性别码(jctb020105):', userData.jctb020105);
      console.log('用户类型(userType):', userData.userType);
      console.log('手机号码(zxxs010109):', userData.zxxs010109);
      console.log('创建时间(createDate):', new Date(userData.createDate).toLocaleString());

      return userData;
    } else {
      throw new Error(`查询失败: ${result.msg}`);
    }
  } catch (error) {
    console.error('调用queryUser接口失败:', error.message);
    throw error;
  }
}

// 调用示例
queryUser('teac')
  .then(data => {
    // 处理用户数据
    console.log('用户姓名:', data.jctb020101);
  })
  .catch(err => {
    // 处理错误
    console.error(err);
  });
```

2. 学生基本信息查询接口

接口描述：批量查询学生基本信息，支持按学年、学期、创建日期筛选，返回学生列表及所在班级信息。

接口地址：`http://{ip}:{port}/oa/api/query/students`

输入参数详解

yearId `String` **可选**

- 说明：学年ID，格式"YYYY-YYYY"，默认当前学年
- 示例：`"2024-2025"`

termId `String` **可选**

- 说明：学期ID，1-第一学期，2-第二学期，默认当前学期
- 示例：`"2"`

createDate `String` **可选**

- 说明：创建日期，格式yyyy-MM-dd，查询该日期之后创建的学生，默认一年前
- 示例：`"2024-09-01"`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，400-客户端错误，501-服务端错误或无数据
- **msg** `String` **必含**：成功时返回学生列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

学生基本信息

- **userId** `String` **必含**：学生登录账号

- **userName** `String` **必含**：学生姓名
- **identityCard** `String` **可选**：身份证号
- **schoolCard** `String` **可选**：学号/学籍号

性别信息

- **gender** `Object` **可选**：性别对象
  - **gender.codeId** `String` **必含**：性别代码（0-未知，1-男性，2-女性，9-未说明）
  - **gender.codeName** `String` **必含**：性别名称（未知的性别、男性、女性、未说明的性别）

班级信息

- **clazz** `Array` **可选**：班级信息数组（一个学生可能有多个班级）
  - **clazz[].clazzTeacher.userId** `String` **必含**：班主任登录账号
  - **clazz[].clazzTeacher.userName** `String` **必含**：班主任姓名
  - **clazz[].clazzId** `Long` **必含**：班级ID
  - **clazz[].clazzName** `String` **必含**：班级名称
  - **clazz[].gradeId** `Long` **必含**：年级ID
  - **clazz[].gradeName** `String` **必含**：年级名称
  - **clazz[].clazzTeacher** `Object` **可选**：班主任信息

学籍状态

- **status** `Object` **可选**：学籍状态对象
  - **status.codeId** `String` **必含**：状态代码，如"00"-注册在籍
  - **status.codeName** `String` **必含**：状态名称，如"注册在籍"

状态码枚举

- `00`：注册在籍
- `01`：公派留学
- `02`：留级
- `03`：降级
- `04`：跳级
- `05`：试读
- `06`：延长年限
- `07`：试读通过
- `08`：回国复学

- `11`：休学
- `12`：复学
- `13`：停学
- `14`：保留入学资格
- `15`：恢复入学资格
- `16`：取消入学资格
- `17`：恢复学籍
- `18`：取消学籍
- `19`：保留学籍
- `21`：转学(转出)
- `22`：转学(转入)
- `23`：转专业
- `24`：专升本
- `25`：本转专
- `26`：转系
- `27`：硕转博
- `28`：博转硕
- `29`：转班级
- `31`：退学
- `42`：开除学籍
- `51`：死亡
- `61`：提前毕业
- `62`：结业
- `63`：肄业
- `64`：国内访学
- `65`：国内访学后返校
- `99`：其他

JavaScript调用示例

```
/**
 * 接口2：查询学生基本信息
 * @param {Object} params - 查询参数
 * @param {string} [params.yearId] - 学年ID，如"2024-2025"
```

```
 * @param {string} [params.termId] - 学期ID，"1"或"2"
 * @param {string} [params.createDate] - 创建日期，yyyy-MM-dd格式
 * @returns {Promise<Array>} 学生信息数组
 */
async function queryStudents(params = {}) {
  const url = 'http://your-server:port/oa/api/query/students';

  // 构建请求参数，使用默认值
  const requestParams = {
    yearId: params.yearId || '',  // 为空则使用系统默认当前学年
    termId: params.termId || '',  // 为空则使用系统默认当前学期
    createDate: params.createDate || '2024-09-01'  // 默认一年前
  };

  // 过滤掉空字符串参数（接口可能期望空字符串或null）
  Object.keys(requestParams).forEach(key => {
    if (requestParams[key] === '') {
      delete requestParams[key];
    }
  });

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(requestParams)
    });

    const result = await response.json();

    // 解析响应
    if (result.code === 200) {
      // msg中是JSON字符串格式的数组
      const students = JSON.parse(result.msg);

      console.log(`成功查询到 ${students.length} 条学生记录`);

      // 遍历输出每条记录的详细信息
      students.forEach((student, index) => {
        console.log(`\n--- 学生 ${index + 1} ---`);
        console.log('用户ID(userId):', student.userId);
        console.log('姓名(userName):', student.userName);
        console.log('身份证号(identityCard):', student.identityCard || '无');
        console.log('学号(schoolCard):', student.schoolCard || '无');
```

```
51
52          // 性别信息
53          if (student.gender) {
54            console.log('性别代码(gender.codeId):', student.gender.codeId);
55            console.log('性别名称(gender.codeName):', student.gender.codeName);
56          }
57
58          // 班级信息（可能多个）
59          if (student.clazz && student.clazz.length > 0) {
60            console.log('班级信息:');
61            student.clazz.forEach((clazz, idx) => {
62              console.log(`  班级${idx+1}: ${clazz.clazzName}(ID:${clazz.clazzId})`);
63              console.log(`    年级: ${clazz.gradeName}(ID:${clazz.gradeId})`);
64              if (clazz.clazzTeacher) {
65                console.log(`    班主任: ${clazz.clazzTeacher.userName}(${clazz.clazzTeacher.userId})`);
66              }
67            });
68          }
69
70          // 学籍状态
71          if (student.status) {
72            console.log('学籍状态代码(status.codeId):', student.status.codeId);
73            console.log('学籍状态名称(status.codeName):', student.status.codeName);
74          }
75        });
76
77        return students;
78      } else {
79        throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
80      }
81    } catch (error) {
82      console.error('调用queryStudents接口失败:', error.message);
83      throw error;
84    }
85  }
86
87  // 调用示例1：使用默认参数（查询最近一年创建的学生）
88  queryStudents()
89    .then(students => {
90      // 处理学生列表
91      console.log('总学生数:', students.length);
92    });
93
94  // 调用示例2：指定学年学期
95  queryStudents({
96    yearId: '2024-2025',
```

```
  termId: '2',
  createDate: '2024-09-01'
});
```

3. 教师基本信息查询接口

接口描述：批量查询教师基本信息，支持按创建日期筛选，返回教师列表及教学相关信息。

接口地址：`http://{ip}:{port}/oa/api/query/staffs`

输入参数详解

createDate `String` **可选**

- 说明：创建日期，格式yyyy-MM-dd，查询该日期之后创建的教师，默认一年前
- 示例：`"2024-09-01"`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，400-客户端错误，501-服务端错误或无数据
- **msg** `String` **必含**：成功时返回教师列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

教师基本信息

- **userId** `String` **必含**：教师登录账号
- **userName** `String` **必含**：教师姓名
- **identityCard** `String` **可选**：身份证号
- **mobile** `String` **可选**：手机号码

- **email** `String` **可选**：电子邮箱

性别信息

- **gender** `Object` **可选**：性别对象
  - **gender.codeId** `String` **必含**：性别代码（0-未知，1-男性，2-女性，9-未说明）
  - **gender.codeName** `String` **必含**：性别名称

教学相关信息

- **subject** `Object` **可选**：任教学科对象
  - **subject.codeId** `String` **必含**：学科代码，如"13"-语文
  - **subject.codeName** `String` **必含**：学科名称，如"语文"
- **schoolPhase** `Object` **可选**：任课学段对象
  - **schoolPhase.codeId** `String` **必含**：学段代码，如"4"-普通高中
  - **schoolPhase.codeName** `String` **必含**：学段名称

个人资质信息

- **education** `Object` **可选**：学历对象
  - **education.codeId** `String` **必含**：学历代码，如"21"-大学本科毕业
  - **education.codeName** `String` **必含**：学历名称
- **jobTitle** `Object` **可选**：职称对象
  - **jobTitle.codeId** `String` **必含**：职称代码，如"052"-高级教师(中学)
  - **jobTitle.codeName** `String` **必含**：职称名称

岗位编制信息

- **staffingLevel** `Object` **可选**：教师编制对象
  - **staffingLevel.codeId** `String` **必含**：编制代码，如"1"-专任教师
  - **staffingLevel.codeName** `String` **必含**：编制名称
- **job** `Object` **可选**：岗位职业对象
  - **job.codeId** `String` **必含**：岗位代码，如"20"-教师兼行政
  - **job.codeName** `String` **必含**：岗位名称

JavaScript调用示例

```
/**
 * 接口3：查询教师基本信息
```

```
 * @param {string} [createDate] - 创建日期，yyyy-MM-dd格式，默认一年前
 * @returns {Promise<Array>} 教师信息数组
 */
async function queryStaffs(createDate = '2024-09-01') {
  const url = 'http://your-server:port/oa/api/query/staffs';

  const requestParams = { createDate };

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(requestParams)
    });

    const result = await response.json();

    if (result.code === 200) {
      const staffs = JSON.parse(result.msg);

      console.log(`成功查询到 ${staffs.length} 条教师记录`);

      staffs.forEach((staff, index) => {
        console.log(`\n--- 教师 ${index + 1} ---`);
        console.log('教师ID(userId):', staff.userId);
        console.log('姓名(userName):', staff.userName);
        console.log('身份证号(identityCard):', staff.identityCard || '无');
        console.log('手机号码(mobile):', staff.mobile || '无');
        console.log('电子邮箱(email):', staff.email || '无');

        // 性别
        if (staff.gender) {
          console.log('性别:', staff.gender.codeName, `(${staff.gender.codeId})`);
        }

        // 任教学科
        if (staff.subject) {
          console.log('任教学科:', staff.subject.codeName, `(${staff.subject.codeId})`);
        }

        // 学历
        if (staff.education) {
          console.log('学历:', staff.education.codeName, `(${staff.education.codeId})`);
        }
```

```
      // 教师编制
      if (staff.staffingLevel) {
        console.log('教师编制:', staff.staffingLevel.codeName, `(${staff.staffingLevel.codeId})`);
      }

      // 岗位职业
      if (staff.job) {
        console.log('岗位职业:', staff.job.codeName, `(${staff.job.codeId})`);
      }

      // 职称
      if (staff.jobTitle) {
        console.log('职称:', staff.jobTitle.codeName, `(${staff.jobTitle.codeId})`);
      }

      // 任课学段
      if (staff.schoolPhase) {
        console.log('任课学段:', staff.schoolPhase.codeName, `(${staff.schoolPhase.codeId})`);
      }
    });

    return staffs;
  } else {
    throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
  }
} catch (error) {
  console.error('调用queryStaffs接口失败:', error.message);
  throw error;
}
}

// 调用示例
queryStaffs('2024-09-01')
  .then(staffs => {
    // 处理教师列表
    const chineseTeachers = staffs.filter(s => s.subject?.codeName === '语文');
    console.log('语文教师数量:', chineseTeachers.length);
  });
```

4. 班级数据查询接口

接口描述：查询班级信息，支持按学年、学期、年级筛选，返回班级详情及班主任、班长信息。

接口地址：`http://{ip}:{port}/oa/api/query/classes`

输入参数详解

yearId `String` **可选**

- 说明：学年ID，格式"YYYY-YYYY"，默认当前学年
- 示例：`"2024-2025"`

termId `String` **可选**

- 说明：学期ID，1-第一学期，2-第二学期，默认当前学期
- 示例：`"2"`

gradeId `Long` **可选**

- 说明：年级ID，如12-高三年级
- 示例：`12`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，400-客户端错误，501-服务端错误或无数据
- **msg** `String` **必含**：成功时返回班级列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

班级基本信息

- **clazzId** `Long` **必含**：班级ID

- **clazzName** `String` **必含**：班级名称
- **gradeId** `Long` **可选**：年级ID
- **gradeName** `String` **可选**：年级名称
- **duration** `String` **可选**：学制，示例：`"三年制"`
- **graduated** `Boolean` **可选**：是否毕业（true/false）

时间信息

- **openingDate** `String` **可选**：入学时间，格式yyyyMM，示例：`"202209"`
- **graduationDate** `String` **可选**：毕业时间，格式yyyyMMdd，示例：`"20250630"`

班级人员

- **clazzTeacher** `Object` **可选**：班主任信息
  - **clazzTeacher.userId** `String` **必含**：班主任登录账号
  - **clazzTeacher.userName** `String` **必含**：班主任姓名
- **clazzLeader** `Object` **可选**：班长信息
  - **clazzLeader.userId** `String` **必含**：班长登录账号
  - **clazzLeader.userName** `String` **必含**：班长姓名

JavaScript调用示例

```javascript
/**
 * 接口4：查询班级数据
 * @param {Object} params - 查询参数
 * @param {string} [params.yearId] - 学年ID，如"2024-2025"
 * @param {string} [params.termId] - 学期ID，"1"或"2"
 * @param {number} [params.gradeId] - 年级ID
 * @returns {Promise<Array>} 班级信息数组
 */
async function queryClasses(params = {}) {
  const url = 'http://your-server:port/oa/api/query/classes';

  const requestParams = {
    yearId: params.yearId || '2024-2025',
    termId: params.termId || '2'
  };

  // 可选参数
  if (params.gradeId) {
    requestParams.gradeId = params.gradeId;
```

```
  }

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(requestParams)
    });

    const result = await response.json();

    if (result.code === 200) {
      const classes = JSON.parse(result.msg);

      console.log(`成功查询到 ${classes.length} 条班级记录`);

      classes.forEach((clazz, index) => {
        console.log(`\n--- 班级 ${index + 1} ---`);
        console.log('班级ID(clazzId):', clazz.clazzId);
        console.log('班级名称(clazzName):', clazz.clazzName);
        console.log('年级ID(gradeId):', clazz.gradeId);
        console.log('年级名称(gradeName):', clazz.gradeName);
        console.log('入学时间(openingDate):', clazz.openingDate || '无');
        console.log('毕业时间(graduationDate):', clazz.graduationDate || '无');

        if (clazz.clazzTeacher) {
          console.log('班主任(clazzTeacher):',
            `${clazz.clazzTeacher.userName}(${clazz.clazzTeacher.userId})`);
        }

        if (clazz.clazzLeader) {
          console.log('班长(clazzLeader):',
            `${clazz.clazzLeader.userName}(${clazz.clazzLeader.userId})`);
        }

        console.log('学制(duration):', clazz.duration || '无');
        console.log('是否毕业(graduated):', clazz.graduated);
      });

      return classes;
    } else {
      throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
    }
  } catch (error) {
```

```
    console.error('调用queryClasses接口失败:', error.message);
    throw error;
  }
}

// 调用示例1：查询当前学年第二学期的所有班级
queryClasses();

// 调用示例2：查询指定年级的班级
queryClasses({
  yearId: '2024-2025',
  termId: '2',
  gradeId: 12  // 高三年级
});
```

5. 学生处分记录查询接口

接口描述：查询学生处分记录，支持按学年、学期、学生筛选，返回处分详情及撤销信息。

接口地址：`http://{ip}:{port}/pspm-sa-sd-sm/api/query/sanctions`

输入参数详解

yearId `String` **可选**

- 说明：学年ID，格式"YYYY-YYYY"，默认当前学年
- 示例：`"2024-2025"`

termId `String` **可选**

- 说明：学期ID，1-第一学期，2-第二学期，默认当前学期
- 示例：`"2"`

userId `String` **可选**

- 说明：学生登录账号，指定查询某学生
- 示例：`"zhangsan"`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，400-客户端错误，501-服务端错误或无数据
- **msg** `String` **必含**：成功时返回处分记录列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

处分基本信息

- **userId** `String` **必含**：学生登录账号
- **item** `String` **必含**：处分名称，示例：`"严重警告"`
- **date** `String` **可选**：处分时间，格式yyyyMMdd，示例：`"20250429"`
- **reason** `String` **可选**：处分原因，示例：`"打架斗殴"`

撤销信息

- **undo** `Boolean` **可选**：是否撤销（true/false）
- **undoReason** `String` **可选**：撤销原因，示例：`"赔礼道歉"`
- **undoDate** `String` **可选**：撤销时间，格式yyyyMMdd，示例：`"20250529"`

JavaScript调用示例

```
/**
 * 接口5：查询学生处分记录
 * @param {Object} params - 查询参数
 * @param {string} [params.yearId] - 学年ID
 * @param {string} [params.termId] - 学期ID
 * @param {string} [params.userId] - 学生ID（可选，不填则查询所有）
 * @returns {Promise<Array>} 处分记录数组
 */
async function querySanctions(params = {}) {
  const url = 'http://your-server:port/pspm-sa-sd-sm/api/query/sanctions';

  const requestParams = {
    yearId: params.yearId || '2024-2025',
    termId: params.termId || '2',
```

```
    userId: params.userId || null
  };

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(requestParams)
    });

    const result = await response.json();

    if (result.code === 200) {
      const sanctions = JSON.parse(result.msg);

      console.log(`成功查询到 ${sanctions.length} 条处分记录`);

      sanctions.forEach((sanction, index) => {
        console.log(`\n--- 处分记录 ${index + 1} ---`);
        console.log('学生ID:', sanction.userId);
        console.log('处分名称:', sanction.item);
        console.log('处分时间:', sanction.date || '无');
        console.log('处分原因:', sanction.reason || '无');
        console.log('是否撤销:', sanction.undo ? '是' : '否');

        if (sanction.undo) {
          console.log('撤销原因:', sanction.undoReason);
          console.log('撤销时间:', sanction.undoDate);
        }
      });

      return sanctions;
    } else {
      throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
    }
  } catch (error) {
    console.error('调用querySanctions接口失败:', error.message);
    throw error;
  }
}

// 调用示例：查询指定学生的处分记录
querySanctions({
  userId: 'zhangsan'
```

```
});
```

6. 学生成绩查询接口

接口描述：查询学生考试成绩，支持按学年、学期、年级、考试筛选，返回详细的成绩数据和排名。

接口地址：`http://{ip}:{port}/pspm-sa-sc/api/query/transcripts`

输入参数详解

yearId `String` **可选**

- 说明：学年ID，格式"YYYY-YYYY"，默认当前学年
- 示例：`"2024-2025"`

termId `String` **可选**

- 说明：学期ID，1-第一学期，2-第二学期，默认当前学期
- 示例：`"2"`

gradeId `Long` **可选**

- 说明：年级ID
- 枚举值：
  - 1：小学一年级
  - 2：小学二年级
  - 3：小学三年级
  - 4：小学四年级
  - 5：小学五年级
  - 6：六年级

- 7：初中一年级
- 8：初中二年级
- 9：初中三年级
- 10：高中一年级
- 11：高中二年级
- 12：高中三年级

testId `Long` **可选**

- 说明：考试ID，指定某次考试
- 示例：`264`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，400-客户端错误，501-服务端错误或无数据
- **msg** `String` **必含**：成功时返回成绩记录列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

考试信息

- **exam** `Object` **必含**：考试信息对象
  - **exam.testId** `Long` **必含**：考试编号
  - **exam.testName** `String` **必含**：考试名称
  - **exam.testType** `Integer` **必含**：考试类型ID
  - **exam.testTypeName** `String` **必含**：考试类型名称

考试类型枚举

- 1：期中考试
- 2：期末考试
- 3：平时考试
- 4：模拟考试
- 5：月考
- 6：单元测试
- 7：区级统考
- 8：市级统考

- 9：省级统考
- 99：其它考试

学科信息

- **course** `Object` **必含**：学科信息对象
  - **course.courseId** `String` **必含**：学科ID
  - **course.courseName** `String` **必含**：学科名称

学生信息

- **student** `Object` **必含**：学生信息对象
  - **student.userId** `String` **必含**：学生登录账号
  - **student.userName** `String` **必含**：学生姓名

成绩信息

- **transcript** `Object` **必含**：成绩信息对象
  - **transcript.score** `Number` **必含**：考分，示例：`125`
  - **transcript.section** `String` **必含**：等第，示例：`"A+"`
  - **transcript.clazzRank** `Integer` **必含**：班级排名，示例：`8`
  - **transcript.gradeRank** `Integer` **必含**：全校/年级排名，示例：`63`
  - **transcript.percent** `Integer` **必含**：全校前百分比，示例：`10`

教师信息

- **teacher** `Object` **可选**：学科教师信息
  - **teacher.userId** `String` **必含**：教师登录账号
  - **teacher.userName** `String` **必含**：教师姓名

JavaScript调用示例

```
/**
 * 接口6：查询学生成绩
 * @param {Object} params - 查询参数
 * @param {string} [params.yearId] - 学年ID
 * @param {string} [params.termId] - 学期ID
 * @param {number} [params.gradeId] - 年级ID
 * @param {number} [params.testId] - 考试ID
 * @returns {Promise<Array>} 成绩记录数组
 */
```

```javascript
async function queryTranscripts(params = {}) {
  const url = 'http://your-server:port/pspm-sa-sc/api/query/transcripts';

  const requestParams = {
    yearId: params.yearId || '2024-2025',
    termId: params.termId || '2'
  };

  if (params.gradeId) requestParams.gradeId = params.gradeId;
  if (params.testId) requestParams.testId = params.testId;

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(requestParams)
    });

    const result = await response.json();

    if (result.code === 200) {
      const transcripts = JSON.parse(result.msg);

      console.log(`成功查询到 ${transcripts.length} 条成绩记录`);

      transcripts.forEach((record, index) => {
        console.log(`\n--- 成绩记录 ${index + 1} ---`);

        // 考试信息
        console.log('考试信息(exam):');
        console.log('  考试ID:', record.exam.testId);
        console.log('  考试名称:', record.exam.testName);
        console.log('  考试类型:', record.exam.testTypeName, `(${record.exam.testType})`);

        // 学科信息
        console.log('学科信息(course):');
        console.log('  学科ID:', record.course.courseId);
        console.log('  学科名称:', record.course.courseName);

        // 学生信息
        console.log('学生信息(student):');
        console.log('  学生ID:', record.student.userId);
        console.log('  学生姓名:', record.student.userName);

```

```
        // 成绩信息
        console.log('成绩信息(transcript):');
        console.log('  考分(score):', record.transcript.score);
        console.log('  等第(section):', record.transcript.section);
        console.log('  班级排名(clazzRank):', record.transcript.clazzRank);
        console.log('  年级排名(gradeRank):', record.transcript.gradeRank);
        console.log('  前百分比(percent):', record.transcript.percent + '%');

        // 教师信息（可选）
        if (record.teacher) {
          console.log('任课教师(teacher):',
            `${record.teacher.userName}(${record.teacher.userId})`);
        }
      });

      return transcripts;
    } else {
      throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
    }
  } catch (error) {
    console.error('调用queryTranscripts接口失败:', error.message);
    throw error;
  }
}

// 调用示例：查询高三年级第二次月考成绩
queryTranscripts({
  yearId: '2024-2025',
  termId: '2',
  gradeId: 12,
  testId: 264
});
```

7. 课表数据查询接口

接口描述：查询课表信息，支持按班级、学生、教师、教室多种维度查询，返回详细的课程安排。

接口地址：`http://{ip}:{port}/pspm-sa-te/api/query/schedules`

输入参数详解

学年学期参数

- **yearId** `String` **可选**：学年ID，格式"YYYY-YYYY"，默认当前学年，示例：`"2024-2025"`
- **termId** `String` **可选**：学期ID，1-第一学期，2-第二学期，默认当前学期，示例：`"2"`

时间筛选参数

- **weekId** `Integer` **可选**：周次，1-20，示例：`1`
- **dayId** `Integer` **可选**：星期，1-7（周一至周日），示例：`5`
- **classNumber** `Integer` **可选**：第几节课，1-9，需配合dayId使用，示例：`1`
- **classDtm** `String` **可选**：上课时间大于等于，格式yyyy-MM-dd HH:mm，示例：`"2025-06-19 08:00"`
- **recessDtm** `String` **可选**：下课时间小于等于，格式yyyy-MM-dd HH:mm，示例：`"2025-06-19 08:45"`

查询维度（四选一，必须提供其中一个）

- **orgId** `Long` **四选一**：班级ID，示例：`1234`
- **studentId** `String` **四选一**：学生登录账号，示例：`"zhangsan"`
- **teacherId** `String` **四选一**：教师登录账号，示例：`"lisi"`
- **roomId** `String` **四选一**：教室ID，示例：`"XJ-1-1"`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，4xx-客户端错误，5xx-服务端错误
- **msg** `String` **必含**：成功时返回课表列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

课表标识

- **scheduleId** `Object` **必含**：课表ID对象
  - **scheduleId.orgId** `Long` **必含**：班级ID
  - **scheduleId.lessonId** `String` **必含**：课堂ID，格式：YYYY-YYYYTAABCC

班级信息

- **clazz** `Object` **必含**：班级信息对象
  - **clazz.classroom.roomId** `String` **必含**：教室ID
  - **clazz.classroom.roomName** `String` **必含**：教室名称
  - **clazz.orgId** `Long` **必含**：班级ID
  - **clazz.orgName** `String` **必含**：班级名称
  - **clazz.classroom** `Object` **必含**：教室信息

课程信息

- **course** `Object` **必含**：课程信息
  - **course.courseId** `String` **必含**：课程ID
  - **course.courseName** `String` **必含**：课程名称

教师信息

- **teacher** `Object` **必含**：任课教师信息
  - **teacher.userId** `String` **必含**：教师登录账号
  - **teacher.userName** `String` **必含**：教师姓名

时间信息

- **time** `Object` **可选**：上课时间信息
  - **time.dayId** `Integer` **必含**：星期几（1-7）
  - **time.dayName** `String` **必含**：星期名称，示例：`"星期五"`
  - **time.classNumber** `Integer` **必含**：第几节课（1-9）
  - **time.classNumberName** `String` **必含**：节次名称，示例：`"第1节"`
  - **time.beginTime** `String` **必含**：上课开始时间，yyyy-MM-dd HH:mm
  - **time.endTime** `String` **必含**：上课结束时间，yyyy-MM-dd HH:mm

JavaScript调用示例

```
/**
 * 接口7：查询课表数据
 * @param {Object} params - 查询参数（必须提供orgId/studentId/teacherId/roomId之一）
 * @returns {Promise<Array>} 课表记录数组
 */
async function querySchedules(params) {
  const url = 'http://your-server:port/pspm-sa-te/api/query/schedules';

```

```
  // 验证必填条件：四个参数至少提供一个
  if (!params.orgId && !params.studentId && !params.teacherId && !params.roomId) {
    throw new Error('必须提供orgId、studentId、teacherId、roomId中的一个');
  }

  const requestParams = {
    yearId: params.yearId || '2024-2025',
    termId: params.termId || '2'
  };

  // 添加可选参数
  if (params.weekId) requestParams.weekId = params.weekId;
  if (params.dayId) requestParams.dayId = params.dayId;
  if (params.classNumber) requestParams.classNumber = params.classNumber;
  if (params.classDtm) requestParams.classDtm = params.classDtm;
  if (params.recessDtm) requestParams.recessDtm = params.recessDtm;

  // 添加查询维度（四选一）
  if (params.orgId) requestParams.orgId = params.orgId;
  if (params.studentId) requestParams.studentId = params.studentId;
  if (params.teacherId) requestParams.teacherId = params.teacherId;
  if (params.roomId) requestParams.roomId = params.roomId;

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(requestParams)
    });

    const result = await response.json();

    if (result.code === 200) {
      const schedules = JSON.parse(result.msg);

      console.log(`成功查询到 ${schedules.length} 条课表记录`);

      schedules.forEach((schedule, index) => {
        console.log(`\n--- 课表记录 ${index + 1} ---`);

        // 课表ID
        console.log('课表ID(scheduleId):');
        console.log('  班级ID(orgId):', schedule.scheduleId.orgId);
        console.log('  课堂ID(lessonId):', schedule.scheduleId.lessonId);
```

```
        // 班级信息
        console.log('班级信息(clazz):');
        console.log('  班级ID:', schedule.clazz.orgId);
        console.log('  班级名称:', schedule.clazz.orgName);
        console.log('  教室ID:', schedule.clazz.classroom.roomId);
        console.log('  教室名称:', schedule.clazz.classroom.roomName);

        // 课程信息
        console.log('课程信息(course):');
        console.log('  课程ID:', schedule.course.courseId);
        console.log('  课程名称:', schedule.course.courseName);

        // 教师信息
        console.log('教师信息(teacher):');
        console.log('  教师ID:', schedule.teacher.userId);
        console.log('  教师姓名:', schedule.teacher.userName);

        // 时间信息
        if (schedule.time) {
          console.log('时间信息(time):');
          console.log('  星期(dayId):', schedule.time.dayId);
          console.log('  星期名称(dayName):', schedule.time.dayName);
          console.log('  节次(classNumber):', schedule.time.classNumber);
          console.log('  节次名称(classNumberName):', schedule.time.classNumberName);
          console.log('  开始时间(beginTime):', schedule.time.beginTime);
          console.log('  结束时间(endTime):', schedule.time.endTime);
        }
      });

      return schedules;
    } else {
      throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
    }
  } catch (error) {
    console.error('调用querySchedules接口失败:', error.message);
    throw error;
  }
}

// 调用示例1：按班级查询课表
querySchedules({
  orgId: 1234,
  yearId: '2024-2025',
  termId: '2'
});
```

```
101
102 // 调用示例2：按教师查询课表
103 querySchedules({
104   teacherId: 'zhangsan',
105   weekId: 1
106 });
107
108 // 调用示例3：按教室查询特定时间的课表
109 querySchedules({
110   roomId: 'XJ-1-1',
111   classDtm: '2025-06-19 08:00',
112   recessDtm: '2025-06-19 08:45'
113 });
```

8. 发布代课需求接口

接口描述：发布代课需求，可直接指定代课教师跳过审核流程，返回发布的代课记录详情。

接口地址：`http://{ip}:{port}/pspm-sa-te/api/substitute/post`

输入参数详解

必填参数

- **lessonId** `String` **必填**：课堂ID，格式：YYYY-YYYYTAABCC，示例：`"2024-2025211101"`
  - 格式说明：YYYY-YYYY(学年) + T(学期) + AA(周次) + B(星期) + CC(节次)
- **classId** `Long` **必填**：班级ID，示例：`123`
- **jctb010403** `String` **必填**：课程ID，示例：`"YW001"`
- **teacherId** `String` **必填**：原配教师ID，示例：`"jw-yw-101"`

可选参数

- **postUserId** `String` **可选**：发布人ID，示例：`"admin"`
- **applyUserId** `String` **可选**：代课教师ID，提供则直接代课成功，示例：`"jw-yw-105"`
- **approveUserId** `String` **可选**：审核人ID，示例：`"admin"`
- **isOfficialTrip** `Boolean` **可选**：是否公务，示例：`true`

输出参数详解

外层响应参数

- **success** `Boolean` **必含**：true-成功，false-失败
- **msg** `String` **必含**：成功时返回data的JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON对象）

基本信息

- **id** `Object` **必含**：课表ID对象
  - **id.lessonId** `String` **必含**：课堂ID
  - **id.classId** `Long` **必含**：班级ID
- **jctb010403** `String` **必含**：课程ID
- **teacherId** `String` **必含**：原配教师ID
- **isOfficialTrip** `Boolean` **可选**：是否公务

发布信息

- **postUserId** `String` **必含**：发布人ID
- **postUserDtm** `Timestamp` **必含**：发布时间（毫秒级时间戳）

审核流程信息

- **lineUserId** `String` **可选**：上线审核人ID
- **lineDtm** `Timestamp` **可选**：上线审核时间
- **applyUserId** `String` **可选**：代课教师ID
- **applyDtm** `Timestamp` **可选**：代课申请时间
- **approveUserId** `String` **可选**：审核人ID
- **approveDtm** `Timestamp` **可选**：审核时间

状态信息

- **status** `Integer` **必含**：代课状态

- 0：待上线
- 1：已下线
- 2：待申请
- 3：待批准
- 4：被驳回
- 5：已批准

JavaScript调用示例

```javascript
/**
 * 接口8：发布代课需求
 * @param {Object} data - 代课需求数据
 * @param {string} data.lessonId - 课堂ID（必填）
 * @param {number} data.classId - 班级ID（必填）
 * @param {string} data.jctb010403 - 课程ID（必填）
 * @param {string} data.teacherId - 原配教师ID（必填）
 * @param {string} [data.postUserId] - 发布人ID
 * @param {string} [data.applyUserId] - 代课教师ID（提供则直接成功）
 * @param {string} [data.approveUserId] - 审核人ID
 * @param {boolean} [data.isOfficialTrip] - 是否公务
 * @returns {Promise<Object>} 发布的代课记录
 */
async function postSubstituteRequest(data) {
  const url = 'http://your-server:port/pspm-sa-te/api/substitute/post';

  // 验证必填参数
  const requiredFields = ['lessonId', 'classId', 'jctb010403', 'teacherId'];
  for (const field of requiredFields) {
    if (!data[field]) {
      throw new Error(`缺少必要参数: ${field}`);
    }
  }

  // 构建请求参数
  const requestParams = {
    lessonId: data.lessonId,
    classId: data.classId,
    jctb010403: data.jctb010403,
    teacherId: data.teacherId
  };

  // 添加可选参数
  if (data.postUserId) requestParams.postUserId = data.postUserId;
```

```
  if (data.applyUserId) requestParams.applyUserId = data.applyUserId;
  if (data.approveUserId) requestParams.approveUserId = data.approveUserId;
  if (data.isOfficialTrip !== undefined) {
    requestParams.isOfficialTrip = data.isOfficialTrip;
  }

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(requestParams)
    });

    const result = await response.json();

    if (result.success === true) {
      const substituteData = JSON.parse(result.msg);

      console.log('代课需求发布成功！');
      console.log('课堂ID(lessonId):', substituteData.id.lessonId);
      console.log('班级ID(classId):', substituteData.id.classId);
      console.log('课程ID(jctb010403):', substituteData.jctb010403);
      console.log('原配教师ID(teacherId):', substituteData.teacherId);
      console.log('发布人ID(postUserId):', substituteData.postUserId);
      console.log('发布时间(postUserDtm):', new Date(substituteData.postUserDtm).toLocaleString());

      // 状态映射
      const statusMap = {
        0: '待上线', 1: '已下线', 2: '待申请',
        3: '待批准', 4: '被驳回', 5: '已批准'
      };
      console.log('代课状态(status):', statusMap[substituteData.status], `(${substituteData.status})`);

      if (substituteData.applyUserId) {
        console.log('代课教师ID(applyUserId):', substituteData.applyUserId);
      }

      if (substituteData.isOfficialTrip !== undefined) {
        console.log('是否公务(isOfficialTrip):', substituteData.isOfficialTrip);
      }

      return substituteData;
    } else {
      throw new Error(`发布失败: ${result.msg}`);
```

```
    }
  } catch (error) {
    console.error('调用postSubstituteRequest接口失败:', error.message);
    throw error;
  }
}

// 调用示例1：发布代课需求（走审核流程）
postSubstituteRequest({
  lessonId: '2024-2025211104',
  classId: 123,
  jctb010403: 'YW001',
  teacherId: 'jw-yw-101',
  postUserId: 'admin',
  isOfficialTrip: true
});

// 调用示例2：直接指定代课教师（跳过审核）
postSubstituteRequest({
  lessonId: '2024-2025211105',
  classId: 123,
  jctb010403: 'YW001',
  teacherId: 'jw-yw-101',
  applyUserId: 'jw-yw-105',  // 直接指定代课教师
  postUserId: 'admin'
});
```

9. 代课数据查询接口

接口描述：查询代课记录，支持多种筛选条件，返回详细的代课信息及审批流程状态。

接口地址：`http://{ip}:{port}/pspm-sa-te/api/query/substitutes`

输入参数详解

学年学期参数

- **yearId** `String` **可选**：学年ID，格式"YYYY-YYYY"，默认当前学年，示例：`"2025-2026"`
- **termId** `String` **可选**：学期ID，1-第一学期，2-第二学期，默认当前学期，示例：`"1"`

时间筛选参数

- **weekId** `Integer` **可选**：周次，1-20，示例：`1`
- **dayId** `Integer` **可选**：星期，1-7，示例：`1`
- **classNumber** `Integer` **可选**：第几节课，1-9，需配合dayId使用，示例：`1`
- **classDtm** `String` **可选**：开始时间，格式yyyy-MM-dd HH:mm，示例：`"2025-09-02 07:20"`
- **recessDtm** `String` **可选**：截止时间，格式yyyy-MM-dd HH:mm，示例：`"2025-09-02 08:00"`

人员筛选参数

- **orgId** `Long` **可选**：班级ID，示例：`199`
- **teacherId** `String` **可选**：原配教师ID，示例：`"jw-yw-101"`
- **applyUserId** `String` **可选**：代课教师ID，示例：`"jw-yw-105"`

状态筛选参数

- **status** `Integer` **可选**：单个代课状态，示例：`5`
- **statuses** `String` **可选**：多个代课状态，逗号分隔，示例：`"1,2,3,4,5"`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，4xx-客户端错误，5xx-服务端错误
- **msg** `String` **必含**：成功时返回代课记录列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

课表标识

- **scheduleId** `Object` **必含**：课表ID对象
  - **scheduleId.orgId** `Long` **必含**：班级ID
  - **scheduleId.lessonId** `String` **必含**：课堂ID

班级信息

- **clazz** `Object` **必含**：班级信息

    - **clazz.classroom.roomId** `String` **必含**：教室ID
    - **clazz.classroom.roomName** `String` **必含**：教室名称
    - **clazz.orgId** `Long` **必含**：班级ID
    - **clazz.orgName** `String` **必含**：班级名称
    - **clazz.classroom** `Object` **必含**：教室信息

课程信息

- **course** `Object` **必含**：课程信息
    - **course.courseId** `String` **必含**：课程ID
    - **course.courseName** `String` **必含**：课程名称

教师信息

- **teacher** `Object` **必含**：原配教师信息
    - **teacher.userId** `String` **必含**：原配教师ID
    - **teacher.userName** `String` **必含**：原配教师姓名

时间信息

- **time** `Object` **必含**：上课时间信息
    - **time.weekId** `Integer` **必含**：周次
    - **time.dayId** `Integer` **必含**：星期几
    - **time.dayName** `String` **必含**：星期名称
    - **time.classNumber** `Integer` **必含**：第几节课
    - **time.classNumberName** `String` **必含**：节次名称
    - **time.beginTime** `String` **必含**：开始时间
    - **time.endTime** `String` **必含**：结束时间

流程信息

- **post** `Object` **必含**：代课发布人信息
    - **post.userId** `String` **必含**：发布人ID
    - **post.userName** `String` **必含**：发布人姓名
    - **post.time** `String` **必含**：发布时间
- **line** `Object` **可选**：代课发布审核人信息
    - **line.userId** `String` **必含**：审核人ID
    - **line.userName** `String` **必含**：审核人姓名

  - **line.time** `String` **必含**：审核时间
- **apply** `Object` **可选**：代课教师信息
  - **apply.userId** `String` **必含**：代课教师ID
  - **apply.userName** `String` **必含**：代课教师姓名
  - **apply.time** `String` **必含**：申请代课时间
- **approve** `Object` **可选**：代课申请审核人信息
  - **approve.userId** `String` **必含**：审核人ID
  - **approve.userName** `String` **必含**：审核人姓名
  - **approve.time** `String` **必含**：审核时间

状态信息

- **status** `Object` **必含**：代课状态对象
  - **status.codeId** `Integer` **必含**：状态代码
  - **status.codeName** `String` **必含**：状态名称
- **official** `Boolean` **可选**：是否公务外出

JavaScript调用示例

```javascript
/**
 * 接口9：查询代课数据
 * @param {Object} params - 查询参数
 * @returns {Promise<Array>} 代课记录数组
 */
async function querySubstitutes(params = {}) {
  const url = 'http://your-server:port/pspm-sa-te/api/query/substitutes';

  const requestParams = {};

  // 添加所有可能的查询参数
  const possibleParams = [
    'yearId', 'termId', 'weekId', 'dayId', 'classNumber',
    'orgId', 'teacherId', 'applyUserId', 'classDtm', 'recessDtm',
    'status', 'statuses'
  ];

  possibleParams.forEach(param => {
    if (params[param] !== undefined && params[param] !== null && params[param] !== '') {
      requestParams[param] = params[param];
    }
  });
```

```
23
24  // 设置默认值（如果没有任何参数）
25  if (Object.keys(requestParams).length === 0) {
26    requestParams.yearId = '2025-2026';
27    requestParams.termId = '1';
28  }
29
30  try {
31    const response = await fetch(url, {
32      method: 'POST',
33      headers: {
34        'Content-Type': 'application/json',
35      },
36      body: JSON.stringify(requestParams)
37    });
38
39    const result = await response.json();
40
41    if (result.code === 200) {
42      const substitutes = JSON.parse(result.msg);
43
44      console.log(`成功查询到 ${substitutes.length} 条代课记录`);
45
46      substitutes.forEach((record, index) => {
47        console.log(`\n--- 代课记录 ${index + 1} ---`);
48
49        // 课表ID
50        console.log('课表ID(scheduleId):', record.scheduleId);
51
52        // 班级信息
53        console.log('班级(clazz):', record.clazz.orgName, `(${record.clazz.orgId})`);
54        console.log('教室:', record.clazz.classroom.roomName);
55
56        // 课程信息
57        console.log('课程(course):', record.course.courseName, `(${record.course.courseId})`);
58
59        // 原配教师
60        console.log('原配教师(teacher):', record.teacher.userName, `(${record.teacher.userId})`);
61
62        // 时间信息
63        console.log('上课时间(time):',
64          `${record.time.dayName} 第${record.time.classNumber}节`,
65          `${record.time.beginTime} 至 ${record.time.endTime}`);
66
67        // 发布人
68        console.log('发布人(post):', record.post.userName, `于 ${record.post.time}`);
```

```
69
70        // 上线审核
71        if (record.line) {
72          console.log('上线审核(line):', record.line.userName, `于 ${record.line.time}`);
73        }
74
75        // 代课教师
76        if (record.apply) {
77          console.log('代课教师(apply):', record.apply.userName, `于 ${record.apply.time}`);
78        }
79
80        // 代课审核
81        if (record.approve) {
82          console.log('代课审核(approve):', record.approve.userName, `于 ${record.approve.time}`);
83        }
84
85        // 状态
86        console.log('状态(status):', record.status.codeName, `(${record.status.codeId})`);
87
88        // 是否公务
89        if (record.official !== undefined) {
90          console.log('是否公务(official):', record.official);
91        }
92        });
93
94        return substitutes;
95      } else {
96        throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
97      }
98    } catch (error) {
99      console.error('调用querySubstitutes接口失败:', error.message);
100     throw error;
101   }
102 }
103
104 // 调用示例1：查询第一周周一的代课记录
105 querySubstitutes({
106   yearId: '2025-2026',
107   termId: '1',
108   weekId: 1,
109   dayId: 1
110 });
111
112 // 调用示例2：查询已批准的代课记录
113 querySubstitutes({
114   status: 5
```

```
});

// 调用示例3：查询多个状态的代课记录
querySubstitutes({
  statuses: '2,3,5'  // 待申请、待批准、已批准
});
```

10. 代课申请被拒数据查询接口

接口描述：查询被驳回的代课申请记录，支持按时间、班级、教师等条件筛选。

接口地址：`http://{ip}:{port}/pspm-sa-te/api/query/substitute/rejected`

输入参数详解

学年学期参数

- **yearId** `String` **可选**：学年ID，格式"YYYY-YYYY"，默认当前学年，示例：`"2025-2026"`
- **termId** `String` **可选**：学期ID，1-第一学期，2-第二学期，默认当前学期，示例：`"1"`

时间筛选参数

- **weekId** `Integer` **可选**：周次，1-20，示例：`1`
- **dayId** `Integer` **可选**：星期，1-7，示例：`1`
- **classNumber** `Integer` **可选**：第几节课，1-9，需配合dayId使用，示例：`1`

人员筛选参数

- **orgId** `Long` **可选**：班级ID，示例：`199`
- **applyUserId** `String` **可选**：代课教师ID，示例：`"jw-yw-105"`
- **approveUserId** `String` **可选**：代课审核教师ID，示例：`"admin"`

其他筛选

- **comment** `String` **可选**：审核备注，模糊查询，示例：`"时间冲突"`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，4xx-客户端错误，5xx-服务端错误
- **msg** `String` **必含**：成功时返回被拒记录列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

课表标识

- **scheduleId** `Object` **必含**：课表ID对象
  - **scheduleId.orgId** `Long` **必含**：班级ID
  - **scheduleId.lessonId** `String` **必含**：课堂ID

班级信息

- **clazz** `Object` **必含**：班级信息
  - **clazz.classroom.roomId** `String` **必含**：教室ID
  - **clazz.classroom.roomName** `String` **必含**：教室名称
  - **clazz.orgId** `Long` **必含**：班级ID
  - **clazz.orgName** `String` **必含**：班级名称
  - **clazz.classroom** `Object` **必含**：教室信息

课程信息

- **course** `Object` **必含**：课程信息
  - **course.courseId** `String` **必含**：课程ID
  - **course.courseName** `String` **必含**：课程名称

教师信息

- **teacher** `Object` **必含**：原配教师信息
  - **teacher.userId** `String` **必含**：原配教师ID
  - **teacher.userName** `String` **必含**：原配教师姓名

时间信息

- **time** `Object` **必含**：上课时间信息
  - **time.dayId** `Integer` **必含**：星期几

    - **time.dayName** `String` **必含**：星期名称
    - **time.classNumber** `Integer` **必含**：第几节课
    - **time.classNumberName** `String` **必含**：节次名称
    - **time.beginTime** `String` **必含**：开始时间
    - **time.endTime** `String` **必含**：结束时间

申请审核信息

- **apply** `Object` **必含**：代课教师信息
    - **apply.userId** `String` **必含**：代课教师ID
    - **apply.userName** `String` **必含**：代课教师姓名
    - **apply.time** `String` **必含**：申请代课时间
- **approve** `Object` **必含**：代课申请审核人信息
    - **approve.userId** `String` **必含**：审核人ID
    - **approve.userName** `String` **必含**：审核人姓名
    - **approve.time** `String` **必含**：审核时间

状态信息

- **status** `Object` **必含**：代课状态对象（固定为被驳回）
    - **status.codeId** `Integer` **必含**：状态代码：4
    - **status.codeName** `String` **必含**：状态名称："被驳回"
- **comment** `String` **可选**：审核备注，驳回原因

JavaScript调用示例

```javascript
/**
 * 接口10：查询代课申请被拒数据
 * @param {Object} params - 查询参数
 * @returns {Promise<Array>} 被拒代课记录数组
 */
async function queryRejectedSubstitutes(params = {}) {
  const url = 'http://your-server:port/pspm-sa-te/api/query/substitute/rejected';

  const requestParams = {};

  // 添加所有可能的查询参数
  const possibleParams = [
    'yearId', 'termId', 'weekId', 'dayId', 'classNumber',
    'orgId', 'applyUserId', 'approveUserId', 'comment'
```

```
];

possibleParams.forEach(param => {
  if (params[param] !== undefined && params[param] !== null && params[param] !== '') {
    requestParams[param] = params[param];
  }
});

// 设置默认值
if (Object.keys(requestParams).length === 0) {
  requestParams.yearId = '2025-2026';
  requestParams.termId = '1';
}

try {
  const response = await fetch(url, {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
    },
    body: JSON.stringify(requestParams)
  });

  const result = await response.json();

  if (result.code === 200) {
    const rejectedRecords = JSON.parse(result.msg);

    if (rejectedRecords.length === 0) {
      console.log('没有查询到被拒的代课记录');
      return [];
    }

    console.log(`成功查询到 ${rejectedRecords.length} 条被拒代课记录`);

    rejectedRecords.forEach((record, index) => {
      console.log(`\n--- 被拒记录 ${index + 1} ---`);

      // 基本信息
      console.log('班级:', record.clazz.orgName);
      console.log('课程:', record.course.courseName);
      console.log('原配教师:', record.teacher.userName);

      // 上课时间
      console.log('上课时间:',
        `${record.time.dayName} 第${record.time.classNumber}节`,
```

```
          `${record.time.beginTime} 至 ${record.time.endTime}`);

        // 代课教师
        console.log('申请代课教师:', record.apply.userName, `于 ${record.apply.time}`);

        // 审核人
        console.log('审核人:', record.approve.userName, `于 ${record.approve.time}`);

        // 驳回原因
        if (record.comment) {
          console.log('驳回原因(comment):', record.comment);
        }

        // 状态
        console.log('状态:', record.status.codeName);
      });

      return rejectedRecords;
    } else if (result.code === 501) {
      console.log('没有数据:', result.msg);
      return [];
    } else {
      throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
    }
  } catch (error) {
    console.error('调用queryRejectedSubstitutes接口失败:', error.message);
    throw error;
  }
}

// 调用示例1：查询第一周的被拒记录
queryRejectedSubstitutes({
  yearId: '2025-2026',
  termId: '1',
  weekId: 1
});

// 调用示例2：查询某教师被拒的记录
queryRejectedSubstitutes({
  applyUserId: 'jw-yw-105'
});

// 调用示例3：按驳回原因模糊查询
queryRejectedSubstitutes({
  comment: '时间冲突'
});
```

11. 课表中的特殊课堂（排课规则）数据查询接口

接口描述：查询排课规则中的特殊课堂，如自习、班会等非普通课程安排。

接口地址：`http://{ip}:{port}/pspm-sa-te/api/query/rules`

输入参数详解

学年学期参数

- **yearId** `String` **可选**：学年ID，格式"YYYY-YYYY"，默认当前学年，示例：`"2025-2026"`
- **termId** `String` **可选**：学期ID，1-第一学期，2-第二学期，默认当前学期，示例：`"1"`
- **weekId** `Integer` **可选**：周次，1-20，默认当前周次，示例：`10`

查询维度（四选一，必须提供其中一个）

- **orgId** `Long` **四选一**：班级ID，示例：`120`
- **studentId** `String` **四选一**：学生登录账号，示例：`""`
- **teacherId** `String` **四选一**：教师登录账号，示例：`""`
- **roomId** `String` **四选一**：教室ID，示例：`""`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，4xx-客户端错误，5xx-服务端错误
- **msg** `String` **必含**：成功时返回特殊课堂列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

时间信息

- **weekId** `Integer` **必含**：上课周/第XX周，1-20

- **dayId** `Integer` **必含**：上课日/星期X，1-7

节次信息

- **lesson** `Object` **必含**：上课节次信息
  - **lesson.classNumber** `Integer` **必含**：节次ID
  - **lesson.classNumberName** `String` **必含**：节次名称，示例：`"第一节晚自"`
  - **lesson.beginTime** `String` **必含**：开始时间，yyyy-MM-dd HH:mm
  - **lesson.endTime** `String` **必含**：结束时间，yyyy-MM-dd HH:mm

课堂类型

- **rule** `String` **必含**：特殊课堂名称，示例：`"班会"`、`"自习"`

JavaScript调用示例

```javascript
/**
 * 接口11：查询特殊课堂（排课规则）
 * @param {Object} params - 查询参数（必须提供orgId/studentId/teacherId/roomId之一）
 * @returns {Promise<Array>} 特殊课堂数组
 */
async function querySpecialRules(params) {
  const url = 'http://your-server:port/pspm-sa-te/api/query/rules';

  // 验证必填条件
  if (!params.orgId && !params.studentId && !params.teacherId && !params.roomId) {
    throw new Error('必须提供orgId、studentId、teacherId、roomId中的一个');
  }

  const requestParams = {
    yearId: params.yearId || '2025-2026',
    termId: params.termId || '1',
    weekId: params.weekId || 10
  };

  // 添加查询维度
  if (params.orgId) requestParams.orgId = params.orgId;
  if (params.studentId) requestParams.studentId = params.studentId;
  if (params.teacherId) requestParams.teacherId = params.teacherId;
  if (params.roomId) requestParams.roomId = params.roomId;

  try {
    const response = await fetch(url, {
      method: 'POST',
```

```
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(requestParams)
    });

    const result = await response.json();

    if (result.code === 200) {
      const rules = JSON.parse(result.msg);

      console.log(`成功查询到 ${rules.length} 条特殊课堂记录`);

      rules.forEach((rule, index) => {
        console.log(`\n--- 特殊课堂 ${index + 1} ---`);
        console.log('周次(weekId):', rule.weekId);
        console.log('星期(dayId):', rule.dayId);
        console.log('节次(classNumber):', rule.lesson.classNumber);
        console.log('节次名称(classNumberName):', rule.lesson.classNumberName);
        console.log('开始时间(beginTime):', rule.lesson.beginTime);
        console.log('结束时间(endTime):', rule.lesson.endTime);
        console.log('特殊课堂名称(rule):', rule.rule);
      });

      return rules;
    } else {
      throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
    }
  } catch (error) {
    console.error('调用querySpecialRules接口失败:', error.message);
    throw error;
  }
}

// 调用示例：查询某班级第10周的特殊课堂
querySpecialRules({
  orgId: 120,
  yearId: '2025-2026',
  termId: '1',
  weekId: 10
});
```

12. 教师周课时查询接口

接口描述：查询教师每周的课时数量，支持按学年、学期、周次筛选。

接口地址：`http://{ip}:{port}/pspm-sa-te/api/query/lessons`

输入参数详解

- **yearId** `String` **可选**：学年ID，格式"YYYY-YYYY"，默认当前学年，示例：`"2024-2025"`
- **termId** `String` **可选**：学期ID，1-第一学期，2-第二学期，默认当前学期，示例：`"2"`
- **weekId** `Integer` **可选**：周次，1-20，示例：`1`
- **teacherId** `String` **可选**：教师登录账号，示例：`"zhangsan"`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，400-客户端错误，501-服务端错误或无数据
- **msg** `String` **必含**：成功时返回教师周课时列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

- **userId** `String` **必含**：教师登录账号
- **lessons** `Array` **必含**：周课时数组
  - **lessons[].weekId** `Integer` **必含**：周次
  - **lessons[].count** `Integer` **必含**：该周课时数量

JavaScript调用示例

```
/**
 * 接口12：查询教师周课时
 * @param {Object} params - 查询参数
```

```
 * @param {string} [params.yearId] - 学年ID
 * @param {string} [params.termId] - 学期ID
 * @param {number} [params.weekId] - 周次
 * @param {string} [params.teacherId] - 教师ID
 * @returns {Promise<Array>} 教师周课时数组
 */
async function queryTeacherLessons(params = {}) {
  const url = 'http://your-server:port/pspm-sa-te/api/query/lessons';

  const requestParams = {
    yearId: params.yearId || '2024-2025',
    termId: params.termId || '2'
  };

  if (params.weekId) requestParams.weekId = params.weekId;
  if (params.teacherId) requestParams.teacherId = params.teacherId;

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(requestParams)
    });

    const result = await response.json();

    if (result.code === 200) {
      const lessonsData = JSON.parse(result.msg);

      console.log(`成功查询到 ${lessonsData.length} 位教师的周课时数据`);

      lessonsData.forEach((teacher, index) => {
        console.log(`\n--- 教师 ${index + 1} ---`);
        console.log('教师ID(userId):', teacher.userId);
        console.log('周课时详情(lessons):');

        // 计算总课时
        let totalLessons = 0;

        teacher.lessons.forEach(lesson => {
          console.log(`  第${lesson.weekId}周: ${lesson.count}节`);
          totalLessons += lesson.count;
        });

```

```
        console.log(`  总计: ${totalLessons}节`);
      });

      return lessonsData;
    } else {
      throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
    }
  } catch (error) {
    console.error('调用queryTeacherLessons接口失败:', error.message);
    throw error;
  }
}

// 调用示例1：查询某教师第1周的课时
queryTeacherLessons({
  teacherId: 'zhangsan',
  weekId: 1
});

// 调用示例2：查询所有教师某学期的周课时
queryTeacherLessons({
  yearId: '2024-2025',
  termId: '2'
});
```

13. 查询指定时间有空（无课）的教师接口

接口描述：查询指定时间段内没有课程安排的教师列表，用于快速找到可代课的教师。

接口地址：`http://{ip}:{port}/pspm-sa-te/api/query/teachers/without-class-schedule`

输入参数详解

必填参数

- **classDtm** `String` **必填**：开始时间，格式yyyy-MM-dd HH:mm，示例：`"2026-03-01 09:30"`
- **recessDtm** `String` **必填**：截止时间，格式yyyy-MM-dd HH:mm，示例：`"2026-03-01 13:30"`

可选参数

- **yearId** `String` **可选**：学年ID，格式"YYYY-YYYY"，默认当前学年，示例：`"2024-2025"`
- **termId** `String` **可选**：学期ID，1-第一学期，2-第二学期，默认当前学期，示例：`"2"`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，400-客户端错误，501-服务端错误或无数据
- **msg** `String` **必含**：成功时返回教师ID数组JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

- **data** `Array` **必含**：教师ID数组，如 `["jw-yy-202", "jw-yy-203"]`

JavaScript调用示例

```javascript
/**
 * 接口13：查询指定时间有空（无课）的教师
 * @param {string} classDtm - 开始时间（必填）
 * @param {string} recessDtm - 截止时间（必填）
 * @param {Object} [options] - 可选参数
 * @param {string} [options.yearId] - 学年ID
 * @param {string} [options.termId] - 学期ID
 * @returns {Promise<Array>} 有空的教师ID数组
 */
async function findAvailableTeachers(classDtm, recessDtm, options = {}) {
  const url = 'http://your-server:port/pspm-sa-te/api/query/teachers/without-class-schedule';

  // 验证必填参数
  if (!classDtm || !recessDtm) {
    throw new Error('classDtm和recessDtm参数是必需的');
  }

  const requestParams = {
    classDtm,
    recessDtm,
    yearId: options.yearId || '2024-2025',
    termId: options.termId || '2'
  };
```

```
  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(requestParams)
    });

    const result = await response.json();

    if (result.code === 200) {
      const teachers = JSON.parse(result.msg);

      console.log(`在时间段 ${classDtm} 至 ${recessDtm} 内，有 ${teachers.length} 位教师有空`);
      console.log('教师ID列表:', teachers);

      return teachers;
    } else {
      throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
    }
  } catch (error) {
    console.error('调用findAvailableTeachers接口失败:', error.message);
    throw error;
  }
}

// 调用示例：查询3月1日上午有空的教师
findAvailableTeachers(
  '2026-03-01 09:30',
  '2026-03-01 13:30',
  { yearId: '2024-2025', termId: '2' }
).then(teacherIds => {
  // 处理教师ID列表
  console.log('可代课教师:', teacherIds.join(', '));
});
```

14. 查询多位教师都有空的时间接口

接口描述：查询多位教师共同空闲的时间段，用于安排会议或集体活动。

接口地址：`http://{ip}:{port}/pspm-sa-te/api/query/lessons/all-have-free-time`

输入参数详解

必填参数

- **teacherIds** `String` **必填**：教师登录账号列表，逗号分隔，示例：`"zhangsan,lisi,wangwu"`

可选参数

- **yearId** `String` **可选**：学年ID，格式"YYYY-YYYY"，默认当前学年，示例：`"2024-2025"`
- **termId** `String` **可选**：学期ID，1-第一学期，2-第二学期，默认当前学期，示例：`"2"`
- **weekId** `Integer` **可选**：周次，1-20，示例：`1`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，400-客户端错误，501-服务端错误或无数据
- **msg** `String` **必含**：成功时返回共同空闲时间列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

- **lessonId** `String` **必含**：课堂ID，格式：YYYY-YYYYTAABCC
- **schoolYearId** `String` **必含**：学年ID，yyyy-yyyy格式
- **termId** `String` **必含**：学期ID，1,2,3,9
- **numberOfWeeks** `Integer` **必含**：周次ID，1-23
- **weekId** `Integer` **必含**：星期ID，1-7
- **classNumber** `Integer` **必含**：节次ID，1-99

- **classDtm** `String` **可选**：上课时间，yyyy-MM-dd HH:mm
- **recessDtm** `String` **可选**：下课时间，yyyy-MM-dd HH:mm

JavaScript调用示例

```javascript
/**
 * 接口14：查询多位教师都有空的时间
 * @param {string|string[]} teacherIds - 教师ID列表（数组或逗号分隔字符串）
 * @param {Object} [options] - 可选参数
 * @param {string} [options.yearId] - 学年ID
 * @param {string} [options.termId] - 学期ID
 * @param {number} [options.weekId] - 周次
 * @returns {Promise<Array>} 共同空闲时间段数组
 */
async function findCommonFreeTime(teacherIds, options = {}) {
  const url = 'http://your-server:port/pspm-sa-te/api/query/lessons/all-have-free-time';

  // 处理teacherIds参数
  let teacherIdsStr;
  if (Array.isArray(teacherIds)) {
    teacherIdsStr = teacherIds.join(',');
  } else {
    teacherIdsStr = teacherIds;
  }

  if (!teacherIdsStr) {
    throw new Error('teacherIds参数是必需的');
  }

  const requestParams = {
    teacherIds: teacherIdsStr,
    yearId: options.yearId || '2024-2025',
    termId: options.termId || '2'
  };

  if (options.weekId) {
    requestParams.weekId = options.weekId;
  }

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
```

```
    body: JSON.stringify(requestParams)
  });

  const result = await response.json();

  if (result.code === 200) {
    const freeTimes = JSON.parse(result.msg);

    console.log(`成功查询到 ${freeTimes.length} 个共同空闲时间段`);

    freeTimes.forEach((time, index) => {
      console.log(`\n--- 空闲时间段 ${index + 1} ---`);
      console.log('课堂ID(lessonId):', time.lessonId);
      console.log('学年:', time.schoolYearId);
      console.log('学期(termId):', time.termId);
      console.log('周次(numberOfWeeks):', time.numberOfWeeks);
      console.log('星期(weekId):', time.weekId);
      console.log('节次(classNumber):', time.classNumber);

      if (time.classDtm) {
        console.log('上课时间(classDtm):', time.classDtm);
        console.log('下课时间(recessDtm):', time.recessDtm);
      }
    });

    return freeTimes;
  } else {
    throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
  }
 } catch (error) {
  console.error('调用findCommonFreeTime接口失败:', error.message);
  throw error;
 }
}

// 调用示例1：使用数组参数
findCommonFreeTime(['zhangsan', 'lisi', 'wangwu'], {
  weekId: 1
});

// 调用示例2：使用字符串参数
findCommonFreeTime('zhangsan,lisi,wangwu', {
  yearId: '2024-2025',
  termId: '2',
  weekId: 1
});
```

15. 教室安排查询接口

接口描述：查询教室安排情况，支持按学年、学期、年级、班级、教室多维度查询。

接口地址：`http://{ip}:{port}/pspm-sa-te/api/query/classrooms`

输入参数详解

- **yearId** `String` **可选**：学年ID，格式"YYYY-YYYY"，默认当前学年，示例：`"2024-2025"`
- **termId** `String` **可选**：学期ID，1-第一学期，2-第二学期，默认当前学期，示例：`"2"`
- **gradeId** `Long` **可选**：年级ID，示例：`2`
- **clazzId** `Long` **可选**：班级ID，示例：`204`
- **roomId** `String` **可选**：房间ID，示例：`"XJ-1-6"`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，400-客户端错误，501-服务端错误或无数据
- **msg** `String` **必含**：成功时返回教室安排列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

- **year** `Object` **必含**：学年信息
  - **year.id** `String` **必含**：学年ID
  - **year.name** `String` **必含**：学年名称
- **term** `Object` **必含**：学期信息
  - **term.id** `String` **必含**：学期ID
  - **term.name** `String` **必含**：学期名称
- **grade** `Object` **必含**：年级信息
  - **grade.id** `Long` **必含**：年级ID

    - **grade.name** `String` **必含**：年级名称
- **clazz** `Object` **必含**：班级信息
    - **clazz.id** `Long` **必含**：班级ID
    - **clazz.name** `String` **必含**：班级名称
- **room** `Object` **必含**：房间信息
    - **room.id** `String` **必含**：房间ID
    - **room.name** `String` **必含**：房间名称

JavaScript调用示例

```javascript
/**
 * 接口15：查询教室安排
 * @param {Object} params - 查询参数
 * @param {string} [params.yearId] - 学年ID
 * @param {string} [params.termId] - 学期ID
 * @param {number} [params.gradeId] - 年级ID
 * @param {number} [params.clazzId] - 班级ID
 * @param {string} [params.roomId] - 房间ID
 * @returns {Promise<Array>} 教室安排数组
 */
async function queryClassrooms(params = {}) {
  const url = 'http://your-server:port/pspm-sa-te/api/query/classrooms';

  const requestParams = {
    yearId: params.yearId || '2024-2025',
    termId: params.termId || '2'
  };

  if (params.gradeId) requestParams.gradeId = params.gradeId;
  if (params.clazzId) requestParams.clazzId = params.clazzId;
  if (params.roomId) requestParams.roomId = params.roomId;

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(requestParams)
    });

    const result = await response.json();
```

```

    if (result.code === 200) {
      const classrooms = JSON.parse(result.msg);

      console.log(`成功查询到 ${classrooms.length} 条教室安排记录`);

      classrooms.forEach((arrangement, index) => {
        console.log(`\n--- 教室安排 ${index + 1} ---`);
        console.log('学年(year):', arrangement.year.name, `(${arrangement.year.id})`);
        console.log('学期(term):', arrangement.term.name, `(${arrangement.term.id})`);
        console.log('年级(grade):', arrangement.grade.name, `(${arrangement.grade.id})`);
        console.log('班级(clazz):', arrangement.clazz.name, `(${arrangement.clazz.id})`);
        console.log('教室(room):', arrangement.room.name, `(${arrangement.room.id})`);
      });

      return classrooms;
    } else {
      throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
    }
  } catch (error) {
    console.error('调用queryClassrooms接口失败:', error.message);
    throw error;
  }
}

// 调用示例1：查询某教室被哪些班级使用
queryClassrooms({
  roomId: 'XJ-1-6',
  yearId: '2025-2026',
  termId: '1'
});

// 调用示例2：查询某年级所有班级的教室安排
queryClassrooms({
  gradeId: 2,
  yearId: '2025-2026',
  termId: '1'
});
```

16. 教师校本课查询接口

接口描述：查询教师开设的校本课程信息，包括课程时间、地点等。

接口地址：`http://{ip}:{port}/pspm-sa-xk/api/query/courses`

输入参数详解

- **status** `Integer` **可选**：选课状态，0-进行中，1-已结束，示例：`0`
- **teacherId** `String` **可选**：教师登录账号，示例：`"zhangsan"`

输出参数详解

外层响应参数

- **code** `Integer` **必含**：200-成功，400-客户端错误，501-服务端错误或无数据
- **msg** `String` **必含**：成功时返回校本课列表JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON数组）

- **userId** `String` **必含**：教师登录账号
- **course** `Object` **必含**：校本课程对象
  - **course.courseId** `Long` **必含**：课程ID
  - **course.courseName** `String` **必含**：课程名称
- **place** `String` **可选**：上课地点，最大64字符，示例：`"XX教室"`
- **startDtm** `String` **必含**：课程时间，最大48字符，示例：`"每周一下午第4节课"`
- **createDtm** `String` **可选**：创建时间，yyyy-MM-dd HH:mm:ss，示例：`"2025-06-09 14:12:11"`
- **modifiedDtm** `String` **可选**：修改时间，yyyy-MM-dd HH:mm:ss，示例：`"2025-06-10 09:30:00"`

JavaScript调用示例

```javascript
/**
 * 接口16：查询教师校本课
 * @param {Object} params - 查询参数
 * @param {number} [params.status] - 选课状态：0-进行中，1-已结束
 * @param {string} [params.teacherId] - 教师ID
 * @returns {Promise<Array>} 校本课数组
 */
async function querySchoolBasedCourses(params = {}) {
  const url = 'http://your-server:port/pspm-sa-xk/api/query/courses';

  const requestParams = {};

  if (params.status !== undefined) {
    requestParams.status = params.status;
  }

  if (params.teacherId) {
    requestParams.teacherId = params.teacherId;
  }

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(requestParams)
    });

    const result = await response.json();

    if (result.code === 200) {
      const courses = JSON.parse(result.msg);

      console.log(`成功查询到 ${courses.length} 条校本课记录`);

      courses.forEach((item, index) => {
        console.log(`\n--- 校本课 ${index + 1} ---`);
        console.log('教师ID(userId):', item.userId);
        console.log('课程ID(courseId):', item.course.courseId);
        console.log('课程名称(courseName):', item.course.courseName);
        console.log('上课地点(place):', item.place || '未指定');
        console.log('课程时间(startDtm):', item.startDtm);
        console.log('创建时间(createDtm):', item.createDtm || '无');
        console.log('修改时间(modifiedDtm):', item.modifiedDtm || '无');
      });
```

```
      return courses;
    } else {
      throw new Error(`查询失败，错误码: ${result.code}, 信息: ${result.msg}`);
    }
  } catch (error) {
    console.error('调用querySchoolBasedCourses接口失败:', error.message);
    throw error;
  }
}

// 调用示例1：查询某教师的所有校本课
querySchoolBasedCourses({
  teacherId: 'zhangsan'
});

// 调用示例2：查询进行中的校本课
querySchoolBasedCourses({
  status: 0
});
```

17. 发布外部资源（JSON参数）接口

接口描述：将音视频外部资源发布到释锐教育资源聚合平台，支持设置资源属性。

接口地址：`http://{ip}:{port}/elearning/api/add/resource`

输入参数详解

请求参数

- **json** `String` **必填**：资源信息描述的JSON字符串

data数据结构（json参数中的对象）

必填参数

- **url** `String` **必填**：音视频网络地址，需支持跨域，示例：`"https://files.heyubao.com/public/warren/文学创思.mp4"`

可选参数

- **title** `String` **可选**：资源标题，不填则从URL截取，示例：`"文学创思"`
- **introduction** `String` **可选**：说明简介，示例：`"资源介绍文本"`
- **isOrginal** `Boolean` **可选**：是否原创，true-是，false-否
- **openDegree** `Integer` **可选**：公开范围，0-私有，1-全校公开，2-全区公开
- **resourceType** `Integer` **可选**：资源类型，0-教学资源，1-专题资源
- **owner** `String` **可选**：资源归属账号，必须有效，示例：`"admin"`
- **status** `Integer` **可选**：审核状态，-1-待审核，0-已审核

输出参数详解

外层响应参数

- **success** `Boolean` **必含**：true-成功，false-失败
- **msg** `String` **必含**：成功时返回data的JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON对象）

基本信息

- **id** `String` **必含**：资源唯一标识
- **title** `String` **必含**：资源标题
- **introduction** `String` **可选**：资源简介
- **previewId** `String` **可选**：外部资源网址，同输入url

资源属性

- **isOrginal** `Boolean` **可选**：是否为原创资源
- **openDegree** `Integer` **必含**：公开范围
- **resourceType** `Integer` **必含**：资源类型
- **isVideo** `Boolean` **可选**：是否是视频
- **videoLength** `Integer` **可选**：时长

统计信息

- **clickCount** `Integer` **可选**：观看次数

- **salesVolume** `Integer` **可选**：销量
- **commentsCount** `Integer` **可选**：评价次数
- **downloadCount** `Integer` **可选**：下载次数
- **citeCount** `Integer` **可选**：引用次数

时间信息

- **uploadDtm** `Datetime` **必含**：上传时间
- **reviewDtm** `Datetime` **可选**：审核时间
- **offShelfDtm** `Datetime` **可选**：上下架日期

人员信息

- **owner** `String` **必含**：上传人
- **reviewer** `String` **可选**：审核人

JavaScript调用示例

```javascript
/**
 * 接口17：发布外部资源（JSON参数）
 * @param {Object} resourceData - 资源数据
 * @param {string} resourceData.url - 音视频URL（必填）
 * @param {string} [resourceData.title] - 资源标题
 * @param {string} [resourceData.introduction] - 资源简介
 * @param {boolean} [resourceData.isOrginal] - 是否原创
 * @param {number} [resourceData.openDegree] - 公开范围
 * @param {number} [resourceData.resourceType] - 资源类型
 * @param {string} [resourceData.owner] - 归属账号
 * @param {number} [resourceData.status] - 审核状态
 * @returns {Promise<Object>} 发布后的资源信息
 */
async function publishResource(resourceData) {
  const url = 'http://your-server:port/elearning/api/add/resource';

  // 验证必填参数
  if (!resourceData.url) {
    throw new Error('url参数是必需的');
  }

  // 构建完整的资源数据
  const fullResourceData = {
    url: resourceData.url,
    title: resourceData.title || '',  // 不填则自动从URL截取
```

```
    introduction: resourceData.introduction || '',
    isOrginal: resourceData.isOrginal !== undefined ? resourceData.isOrginal : true,
    openDegree: resourceData.openDegree !== undefined ? resourceData.openDegree : 1,  // 默认全校公开
    resourceType: resourceData.resourceType !== undefined ? resourceData.resourceType : 0,  // 默认教学资源
    owner: resourceData.owner || 'admin',
    status: resourceData.status !== undefined ? resourceData.status : -1  // 默认待审核
  };

  try {
    const response = await fetch(url, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/x-www-form-urlencoded',
      },
      body: new URLSearchParams({
        json: JSON.stringify(fullResourceData)
      })
    });

    const result = await response.json();

    if (result.success === true) {
      const publishedResource = JSON.parse(result.msg);

      console.log('资源发布成功! ');
      console.log('资源ID(id):', publishedResource.id);
      console.log('标题(title):', publishedResource.title);
      console.log('简介(introduction):', publishedResource.introduction || '无');
      console.log('URL(previewId):', publishedResource.previewId);
      console.log('上传人(owner):', publishedResource.owner);
      console.log('上传时间(uploadDtm):', publishedResource.uploadDtm);
      console.log('公开范围(openDegree):', publishedResource.openDegree);
      console.log('资源类型(resourceType):', publishedResource.resourceType);
      console.log('是否原创(isOrginal):', publishedResource.isOrginal);

      return publishedResource;
    } else {
      throw new Error(`发布失败: ${result.msg}`);
    }
  } catch (error) {
    console.error('调用publishResource接口失败:', error.message);
    throw error;
  }
}

// 调用示例
```

```
publishResource({
  url: 'https://files.heyubao.com/public/warren/文学创思.mp4?key=hello&key2=world',
  title: '文学创思',
  introduction: '这是一部关于文学创作的思考视频',
  isOrginal: true,
  openDegree: 1,  // 全校公开
  resourceType: 0,  // 教学资源
  owner: 'admin',
  status: -1  // 待审核
});
```

18. 发布外部资源（简易参数）接口

接口描述：使用简化参数快速发布音视频外部资源，只需提供URL即可。

接口地址：`http://{ip}:{port}/elearning/api/add/resource/tiny`

输入参数详解

必填参数

- **url** `String` **必填**：音视频网络地址，需支持跨域，示例：`"https://files.heyubao.com/public/warren/文学创思.mp4"`

可选参数

- **title** `String` **可选**：资源标题，不填则从URL截取，示例：`"文学创思"`
- **owner** `String` **可选**：资源归属账号，必须有效，默认`"admin"`，示例：`"admin"`

输出参数详解

外层响应参数

- **success** `Boolean` **必含**：true-成功，false-失败

- **msg** `String` **必含**：成功时返回data的JSON字符串，失败时返回错误信息

data数据结构（msg中的JSON对象）

与接口17的data结构完全相同，包含资源的所有详细信息。

JavaScript调用示例

```javascript
/**
 * 接口18：发布外部资源（简易参数）
 * @param {string} url - 音视频URL（必填）
 * @param {string} [title] - 资源标题
 * @param {string} [owner] - 归属账号，默认admin
 * @returns {Promise<Object>} 发布后的资源信息
 */
async function publishResourceSimple(url, title = '', owner = 'admin') {
  const apiUrl = 'http://your-server:port/elearning/api/add/resource/tiny';

  // 验证必填参数
  if (!url) {
    throw new Error('url参数是必需的');
  }

  const params = new URLSearchParams({ url });

  if (title) {
    params.append('title', title);
  }

  if (owner) {
    params.append('owner', owner);
  }

  try {
    const response = await fetch(apiUrl, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/x-www-form-urlencoded',
      },
      body: params
    });

    const result = await response.json();

    if (result.success === true) {
```

```
      const publishedResource = JSON.parse(result.msg);

      console.log('资源发布成功（简易模式）！');
      console.log('资源ID(id):', publishedResource.id);
      console.log('标题(title):', publishedResource.title);
      console.log('URL(previewId):', publishedResource.previewId);
      console.log('上传人(owner):', publishedResource.owner);
      console.log('上传时间(uploadDtm):', publishedResource.uploadDtm);

      return publishedResource;
    } else {
      throw new Error(`发布失败: ${result.msg}`);
    }
  } catch (error) {
    console.error('调用publishResourceSimple接口失败:', error.message);
    throw error;
  }
}

// 调用示例1：只提供URL
publishResourceSimple('https://files.heyubao.com/public/warren/文学创思.mp4');

// 调用示例2：提供完整信息
publishResourceSimple(
  'https://files.heyubao.com/public/warren/数学思维训练.mp4',
  '数学思维训练',
  'teacher_wang'
);
```

通用工具函数

为了方便调用，提供以下通用工具函数：

```
/**
 * 安全调用API的包装函数
 * @param {Function} apiFunction - API调用函数
 * @param  {...any} args - 函数参数
 */
async function safeApiCall(apiFunction, ...args) {
  try {
    console.log(`调用API: ${apiFunction.name}`);
    const startTime = Date.now();
```

```
    const result = await apiFunction(...args);

    const elapsedTime = Date.now() - startTime;
    console.log(`API调用成功，耗时: ${elapsedTime}ms`);

    return result;
  } catch (error) {
    console.error(`API调用失败: ${error.message}`);
    throw error;
  }
}

/**
 * 从URL提取文件标题
 * @param {string} url - 文件URL
 * @returns {string} 标题
 */
function extractTitleFromUrl(url) {
  try {
    const urlObj = new URL(url);
    const filename = urlObj.pathname.split('/').pop();
    return decodeURIComponent(filename.replace(/\.[^/.]+$/, ''));
  } catch (error) {
    return '未命名资源';
  }
}

// 使用示例
safeApiCall(queryUser, 'teac')
  .then(userData => {
    console.log('用户数据获取成功');
  });
```

注意事项

IP白名单：所有接口调用都需要将调用方IP加入释锐系统白名单

跨域问题：如果从浏览器前端直接调用，需要服务器支持CORS

参数格式：

- 时间格式严格按接口要求（yyyy-MM-dd、yyyyMMdd、timestamp等）
- 布尔值使用true/false（小写）

- 枚举值使用接口定义的代码

必填验证：注意各接口的必填参数，特别是四选一参数组

响应处理：

- 部分接口的msg中是字符串化的JSON，需要二次解析
- 注意区分success+msg和code+msg两种响应格式

模块依赖：必须购买相应应用模块方可调用对应API

文档版本：v1.8
更新日期：2026年03月03日

技术支持：请访问公司官网联系交付与售后团队