

Interactive OLX Parser User Guide

Overview

The Interactive OLX Parser is a powerful Python tool that transforms edX OLX (Open Learning XML) course structures into beautiful, searchable, interactive HTML visualizations. Whether you're working with course directories, zip files, or compressed archives, this tool makes it easy to explore and understand complex course hierarchies.

Quick Start

Installation Requirements

- Python 3.6 or higher
- No additional dependencies required (uses only Python standard library)

Basic Usage

```
bash
```

```
python olx_parser.py /path/to/your/course
```

The parser will automatically generate an HTML file named `{course_name}_structure.html` that you can open in any web browser.

Supported Input Formats

The parser accepts multiple input formats:

| Format | Example | Description |
|-----------|-----------------------------------|--------------------------|
| Directory | <code>./my_course/</code> | Raw OLX course directory |
| ZIP File | <code>course_export.zip</code> | Compressed course export |
| TAR.GZ | <code>course_backup.tar.gz</code> | Gzipped tar archive |
| TGZ | <code>course_data.tgz</code> | Alternative gzip format |

Example Commands

```
bash
```

```
# Parse a course directory
```

```
python olx_parser.py ./edx_course_directory/
```

```
# Parse a zip file
```

```
python olx_parser.py course_export.zip
```

```
# Parse with verbose output
```

```
python olx_parser.py -v course_backup.tar.gz
```

```
# Specify custom output filename
```

```
python olx_parser.py -o my_custom_report.html ./course/
```

Command Line Options

Required Arguments

- **olx_path**: Path to your OLX course (directory, zip, tar.gz, or tgz file)

Optional Arguments

- **-v, --verbose**: Enable detailed debug output during parsing
- **-o, --output**: Specify custom output HTML filename
- **-h, --help**: Show help message and exit

Examples

```
bash
```

```
# Verbose parsing with custom output
```

```
python olx_parser.py -v -o detailed_analysis.html course.zip
```

```
# Quick parsing with default settings
```

```
python olx_parser.py my_course_folder/
```

Interactive HTML Features

Search Functionality

The generated HTML includes powerful search capabilities:

- **Global Search**: Search across all component names, types, and URL identifiers
- **Real-time Results**: See results as you type with 300ms debounce

- **Smart Highlighting:** Matching text is highlighted in yellow
- **Auto-expansion:** Parent nodes automatically expand to show matches

Search Examples

- Search `"video"` to find all video components
- Search `"problem"` to locate assessment items
- Search `"week"` to find weekly content sections
- Search specific component names or URL identifiers

Tree Navigation

Navigate your course structure intuitively:









- **Expandable Nodes:** Click `[+]` to expand, `[-]` to collapse
- **Smart Icons:** Each component type has a unique emoji identifier
- **Hierarchical View:** Clear parent-child relationships
- **Bulk Controls:** "Expand All" and "Collapse All" buttons

Component Summary

Get an instant overview of your course:

- **Visual Statistics:** Grid layout showing component counts
- **Icon-coded Types:** Quick visual identification
- **Total Counts:** See the full scope of your course content

Component Type Icons

| Type | Icon | Description |
|------------|---|-----------------------------|
| Course |  | Root course container |
| Chapter |  | Major course sections |
| Sequential |  | Sequential content units |
| Vertical |  | Vertical content containers |
| Problem |  | Assessment components |
| Video |  | Video content |
| HTML |  | HTML text content |
| Discussion |  | Discussion forums |

Troubleshooting

Common Issues

"Could not find OLX course structure"

- **Cause:** Missing `course.xml` file in root directory
- **Solution:** Ensure your OLX export is complete and contains the required root files

"File not found" errors during parsing

- **Cause:** Incomplete or corrupted OLX export
- **Solution:** Re-export your course or check file integrity

Empty or missing components

- **Cause:** Broken references in XML files
- **Solution:** Use verbose mode (`-v`) to identify specific parsing issues

Debug Mode

Use the `-v` flag to see detailed parsing information:

```
bash  
  
python olx_parser.py -v your_course.zip
```

This will show:

- Extraction progress for compressed files
- OLX structure discovery process
- Component parsing details
- File path resolution steps

File Permission Issues

If you encounter permission errors:

- Ensure read access to input files
- Check write permissions for output directory
- Run with appropriate user privileges

Best Practices

Course Organization

- Use descriptive `display_name` attributes in your OLX files
- Maintain consistent naming conventions
- Keep component hierarchies logical and shallow when possible

Performance Tips

- For very large courses (1000+ components), consider parsing specific sections
- Use descriptive output filenames for multiple course analyses
- Clean up temporary files if parsing many compressed archives

Workflow Integration

1. **Course Development:** Use during development to visualize structure
2. **QA Testing:** Verify course organization before publication
3. **Documentation:** Generate visual course maps for stakeholders
4. **Migration:** Analyze legacy courses before platform transitions

Advanced Usage

Batch Processing

Process multiple courses efficiently:

```
bash

# Process all zip files in a directory
for file in *.zip; do
    python olx_parser.py "$file"
done
```

Custom Output Organization

```
bash

# Create organized output directory
mkdir course_analyses
python olx_parser.py -o "course_analyses/$(date +%Y%m%d)_analysis.html" course.zip
```

Integration with Course Development

- Add to your course build pipeline
- Use for automated documentation generation
- Include in quality assurance workflows



Output File Structure

The generated HTML file is self-contained and includes:

- **Complete course hierarchy:** All components and relationships
- **Interactive JavaScript:** Search and navigation functionality
- **Embedded CSS:** Professional styling and animations
- **Browser compatibility:** Works in all modern browsers
- **No external dependencies:** Fully portable HTML file



Use Cases

For Course Authors

- **Structure Planning:** Visualize course organization before development
- **Content Auditing:** Identify missing or misplaced components
- **Collaboration:** Share visual course maps with team members

For Platform Administrators

- **Course Analysis:** Understand course complexity and structure
- **Migration Planning:** Assess courses before platform transitions
- **Quality Assurance:** Verify course exports and imports

For Developers

- **OLX Understanding:** Learn OLX file structure and relationships
- **Debugging:** Identify structural issues in course exports
- **Tool Development:** Use as reference for OLX processing tools



Limitations

- **Read-only Analysis:** Does not modify or validate OLX content
- **Structure Focus:** Concentrates on hierarchy, not content quality
- **Memory Usage:** Very large courses may require significant RAM

- **OLX Variants:** Optimized for standard edX OLX format



Additional Resources

- **edX OLX Documentation:** [Official edX OLX Guide](#)
- **Course Export Guide:** edX Studio Course Export documentation
- **OLX Best Practices:** Community guidelines for OLX organization



Support

If you encounter issues:

1. **Check the console output** for error messages
2. **Use verbose mode** (`(-v)`) for detailed debugging
3. **Verify your OLX structure** has required files (`course.xml`), (`/course/` directory)
4. **Test with a simple course** to isolate complex structure issues

Happy course parsing! 🇮🇹 ✨