RPA Design and Development

v4.0

UiPath

Lesson 5 | Workflow Analyzer

Workflow Analyzer– Exam Topics

- ❑ Use Workflow Analysis and Validation at file and project level
- ❑ Configure Workflow Analyzer settings

Learning Objectives

1. Explain what is Workflow Analyzer and how it works
2. Describe the Workflow Analyzer rules categories and components
3. Use the Error List panel to find errors within the workflow by using the description and additional resources to solve them
4. Execute an analysis of your project to check for validation errors, any quality and reliability standards set as rules in the Workflow Analyzer
5. Locate and edit the Workflow Analyzer rules as per project requirements
6. Ensure to follow general set of best practices in project

Workflow Analyzer

Workflow Analyzer is a **static code analyzer** that ensures your project meets high quality and reliability standards.

A static code analyzer

- **checks for inconsistencies without actually executing the project**, as opposed to dynamic analyzers which step in during execution
- **enforce automation best practices**
- **ensures high quality and reliability standards**

Introduction to Workflow Analyzer

- Uses a set of rules to check for various inconsistencies unrelated to project execution
- The rules are based on automation best practices and take into consideration variable and argument naming, empty sequences or workflows, package restrictions, and so on
- does not identify errors during execution or compilation

What is Workflow Analyzer?

Workflow Analyzer

- Is a static code analyzer that ensures your project meets high quality and reliability standards
- Uses the configured rule set to check the project or workflow for inconsistencies without executing the project.
- It logs the errors found in the Error List panel, under the rule action (Error, Warning, Info or Verbose).
- The rules are based on UiPath Automation Best Practices

Using the Workflow Analyzer, you can

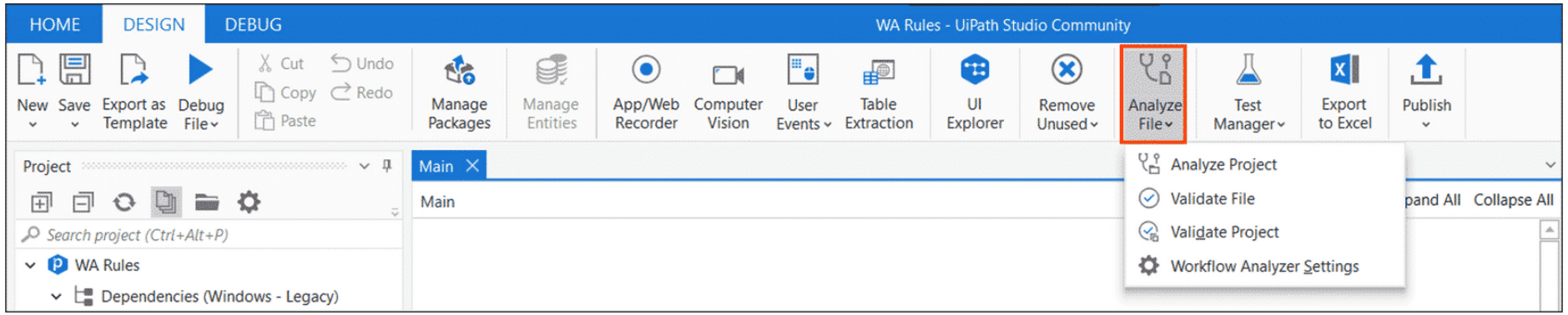
- Edit, disable and enable rules from Studio
- Run validation or validation and analysis at file or project level
- Manage errors and warnings in the Error List panel
- Build custom rules using the **UiPath.Activities.Api package**
- Integrate workflow analysis with prebuilt and/or custom rules in CI/CD ipeline configurations.

Why you need Workflow Analyzer?

Workflow analysis plays a central part in project development.

Workflow Analyzer is a tool for

- making sure your project follows best practices
- ensuring higher quality, reliability and readability



You can access the Workflow Analyzer options by clicking the **Analyze File drop-down menu** from the Design ribbon tab.

Running workflow analysis is a **two step process**.

1. The Analyzer runs workflow validation (checks for compilation errors) first, and
2. only after it passes, it analyzes the workflow (checks for breaches against the defined Workflow Analyzer rules)

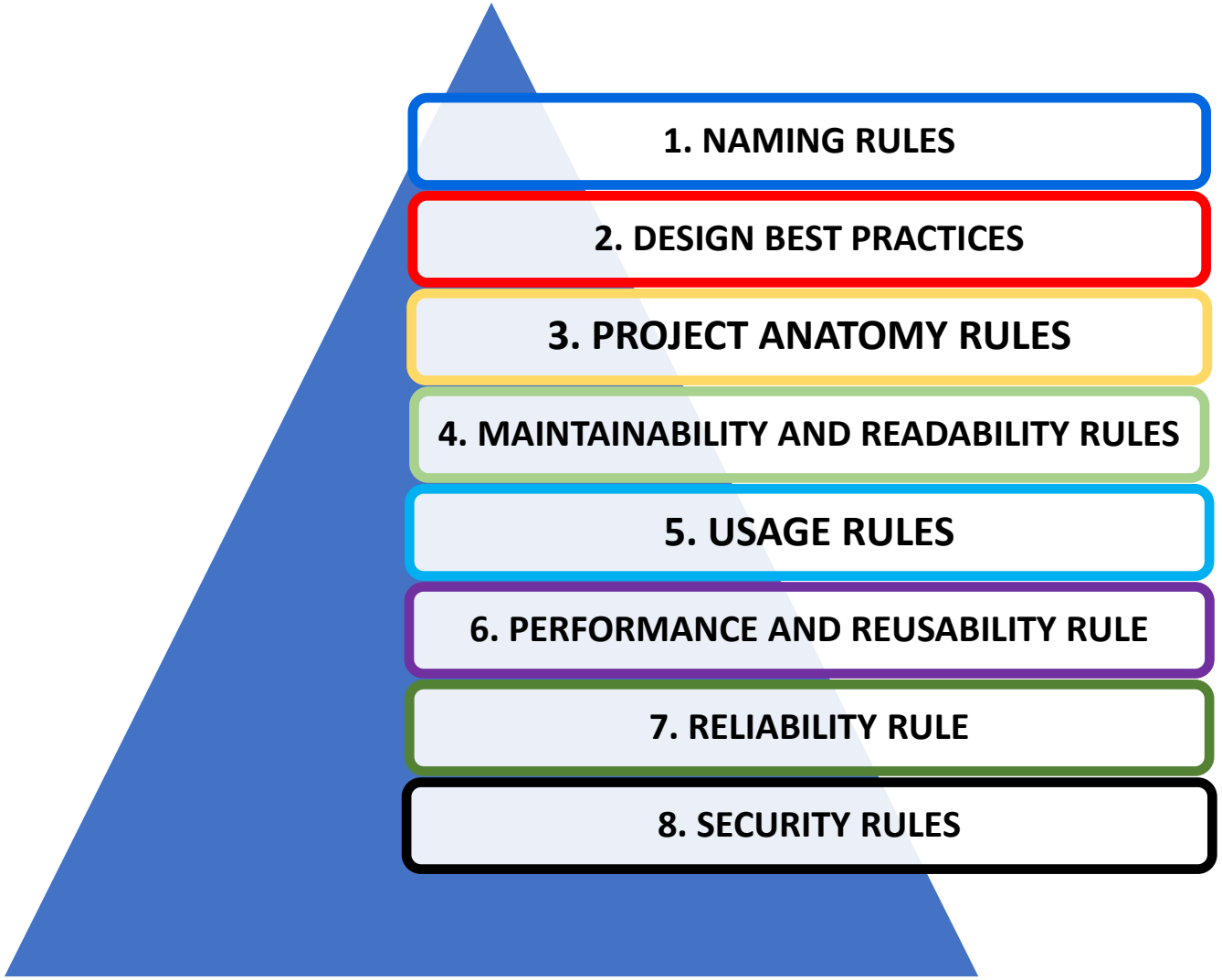
Enforcing Workflow Analyzer

The workflow analyzer can be enforced using following ways:

To enable these options, access **Home > Settings > Design**

Enforce Option	Description
Enforce Analyzer before Run	This option runs the Workflow Analyzer before running or debugging a workflow file and checks for all the rules with Error action. It allows execution of the workflow only if no errors are found.
Enforce Analyzer before Publish	This option runs the Workflow Analyzer before publishing a workflow file or a project. It checks for all the enabled rules regardless of their action and allows publishing only if there are no rule violations with the action Error.
Enforce Analyzer before Push/Check In	Whenever sending a project to a remote repository is initiated (Commit and Push for GIT, Check in for SVN and TFS), the Workflow Analyzer checks all enabled rules regardless of their action and the operation is allowed only if there are no rule violations with the action Error. if publishing is successful (there are no rule violations with the action Error) the results of the workflow analysis **are included in the published .nupkg package** in the file **project_analysis_results.json** located in \lib\net45\.analysis\.

Workflow Analyzer rules categories



Rules Categories : NAMING RULES

Rules in this category carry the **NMG** code in their ID.

This category checks the file or project for **any inconsistencies related to naming**.

Below is the list of rules:

Rule Name	Rule ID	Scope
Variables Naming Convention	ST-NMG-001	Activity
Arguments Naming Convention	ST-NMG-002	Workflow
Display Name Duplication	ST-NMG-004	Workflow
Variable Overrides Variable	ST-NMG-005	Workflow
Variable Overrides Argument	ST-NMG-006	Workflow
Variable Length Exceeded	ST-NMG-008	Activity
Prefix Datatable Variables	ST-NMG-009	Activity
Prefix Datatable Arguments*	ST-NMG-011	Workflow
Argument Default Values	ST-NMG-012	Workflow
Argument Length Exceeded	ST-NMG-016	Workflow

Rules Categories : DESIGN BEST PRACTICES

Rules in this category contain the **DBP** code in their ID.

This category refers to requirements for **ensuring your project meets a general set of best practices**.

Below is the list of rules:

Rule Name	Rule ID	Scope
High Arguments Count	ST-DBP-002	Workflow
Empty Catch Block	ST-DBP-003	Activity
Multiple Flowchart Layers	ST-DBP-007	Workflow
Undefined Output Properties	ST-DBP-020	Activity
Empty Workflow	ST-DBP-023	Workflow
Persistence Activity Check	ST-DBP-024	Workflow
Variables Serialization Prerequisite	ST-DBP-025	Workflow
Delay Activity Usage	ST-DBP-026	Workflow
Persistence Best Practice	ST-DBP-027	Workflow
Arguments Serialization Prerequisite	ST-DBP-028	Workflow

Rules Categories : PROJECT ANATOMY RULES

Rules in this category contain **ANA** code in their ID.

The rules in this category ensure your project **meets general requirements in terms of anatomy**.

Below is the list of rules:

Rule Name	Rule ID	Scope
Project Workflow Count	ST-ANA-003	Workflow
Check Project.json Exists	ST-ANA-005	Project
Main Workflow Exists	ST-ANA-006	Project
File Activities Stats	ST-ANA-009	Activity

Rules Categories :
MAINTAINABILITY AND READABILITY RULES

Rules in this category contain **MRD** code in their rule ID.

The rules in this category **require projects to be easy to understand so that maintainability is ensured**.

Below is the list of rules:

Rule Name	Rule ID	Scope
Activity Name Defaults	ST-MRD-002	Activity
Unreachable Activities	ST-MRD-004	Workflow
Redundant Sequences	ST-MRD-005	Workflow
Nested If Clauses	ST-MRD-007	Workflow
Empty Sequence	ST-MRD-008	Workflow
Deeply Nested Activities	ST-MRD-009	Workflow
Write Line Usage	ST-MRD-011	Workflow
Incomplete If	ST-MRD-017	Activity

Rules Categories : USAGE RULES

Rules in this category contain **USG** code in their ID.

The rules in this category refer to requirements **for ensuring elements defined in your project are actually used.**

Below is the list of rules:

Rule Name	Rule ID	Scope
Hardcoded Activity Arguments	ST-USG-005	Activity
Unused Variables	ST-USG-009	Workflow
Unused Dependencies	ST-USG-010	Project
Package Restrictions	ST-USG-014	Project
Minimum Log Messages	ST-USG-020	Workflow
Unused Saved for Later	ST-USG-024	Workflow
Saved Value Misuse	ST-USG-025	Workflow
Activity Restrictions	ST-USG-026	Workflow
Required Packages	ST-USG-027	Project

Rules Categories :
PERFORMANCE AND REUSABILITY RULE

A rule in this category contains **PRR** code in its ID.

The rule in this category checks the file or project for **any inconsistencies related to performance and reusability**.

Below is the rule:

Rule Name	Rule ID	Scope
Hardcoded Delay Activity	ST-PRR-004	Workflow

Rules Categories : RELIABILITY RULE

A rule in this category contains **REL** code in its ID.

The rule is in this category checks the file or project for **any inconsistencies related to reliability**.

Below is the rule:

Rule Name	Rule ID	Scope
Infinite Loop	ST-REL-006	Activity

Rules Categories : SECURITY RULES

The rules in this category contain **SEC** code in their ID.

The rules in this category check the file or project for **any inconsistencies related to security**.

Below is the list of rules:

Rule Name	Rule ID	Scope
SecureString Argument Usage	ST-SEC-007	Workflow
SecureString Variable Usage	ST-SEC-008	Workflow
SecureString Misusage	ST-SEC-009	Workflow

Rules Categories

Note that certain activity packages such as Excel, Mail, Testing, and UI Automation **will come with their own workflow analyzer rules**.

These rules will get added automatically when your workflow uses any of these activity packages.

Various releases may update the categories and rules for each category.

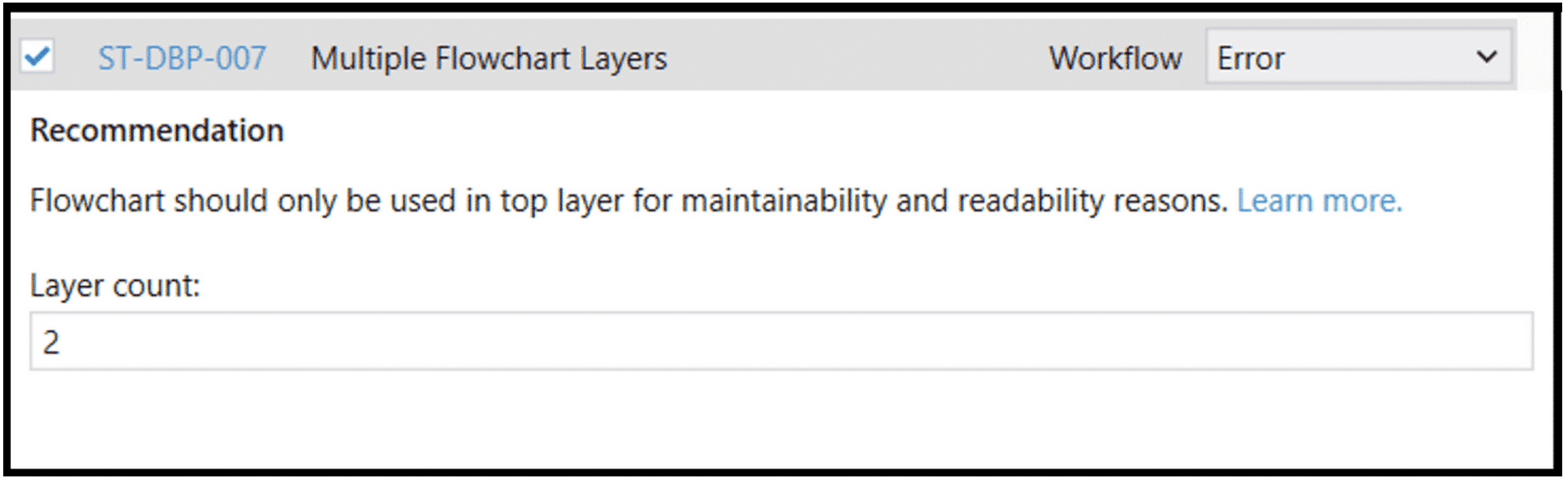
Stay up-to-date by reviewing the documentation in the Workflow Analyzer section of the UiPath Studio Guide.

Workflow Analyzer Rules Components

Each rule has an

- **ID**
- **Name**
- **Description**
- **Recommendation scope**
- **Default action**
- **And some rules have an editable property**

Note that if you have changed any default parameters and want to revert then right-click the selected rule and select Reset to Default option



Exporting Workflow Analyzer results

You can configure Studio to export the results of each workflow analysis to the project folder.

Go to Studio **Backstage View > Settings > Design** and **enable the option Export Analyzer results**.

When this option is enabled, the results of each workflow analysis are saved in the JSON format in the \.local\.analysis\ subfolder of the project folder.

The file will contain the following information about each enabled rule:

- RuleId
- RuleName
- Parameters
- Severity
- ErrorsDescription

Note: The .local folder is hidden. You can enable viewing hidden items from the Windows File Explorer settings.

Resources

Topic	Link
About Workflow Analyzer	https://docs.uipath.com/studio/standalone/2022.10/user-guide/about-workflow-analyzer
Design Best Practices	https://docs.uipath.com/studio/standalone/2022.10/user-guide/design-best-practices
Workflow Analyzer Rules	https://docs.uipath.com/activities/other/latest/ui-automation/ui-automation-workflow-analyzer-rules
Building Workflow Analyzer Rules	https://docs.uipath.com/activities/other/latest/developer/building-workflow-analyzer-rules

Using the Workflow Analyzer in Studio

The Workflow Analyzer is an essential tool that helps detect and address discrepancies in a project or file.

- It utilizes a robust collection of rules **to ensure that any issues are identified and resolved before proceeding with any subsequent operations**, such as debugging, executing, or publishing
- These **rules can be further refined and customized** to suit the specific project by selecting or deselecting the corresponding options
- There are **varying levels of operation for each rules**, which **can be personalized** in accordance with industry standards and best practices

A few best practices includes

- If two variables with the same name exist, although we highly recommend against it, the one defined in the most inner scope has priority
- You can only assign one default case. Default cases are not numbered
- Reduce the number of arguments in a workflow to a minimum
- It is recommended to insert log messages in the Catch block of a Try Catch activity, in addition to the exception handling itself
- It is recommended to reduce the number of flowchart layers to a minimum to ensure maintainability and readability
- Make sure to exclude empty files from the project in order to keep only relevant workflows and to reduce the total size of the project
- Use Wait ... and Resume persistence activities only in the workflow file that is set as Main
- Do not use the Delay activity in the workflow file set as Main in projects that support persistence. Either move Delay activities to other workflow files in the project or use the Resume After Delay activity in the file set as Main
- Change the argument data type to a serializable data type

Advanced Features of Workflow Analyzer

Command-line user interface contains a set of parameters for checking files or projects against certain rules, even in CI/CD pipeline configurations

To ensure Studio users adhere to development standards and comply with certain rules, organizations can enforce governance policies that control Studio functionalities.

In its off-the-shelf form, Workflow Analyzer comes with a wide set of rules

What is Command-Line support?

The **UiPath.Studio.CommandLine.exe** command-line user interface contains a set of parameters for checking files or projects against certain rules, even in CI/CD pipeline configurations.

Where is it available?

UiPath.Studio.CommandLine.exe is available in the installation folder:

- For the Windows Installer (.msi) version of Studio, the default path is %ProgramFiles%\UiPath\Studio\
- For Studio installed in User Mode (.exe), the default path is %LocalAppData%\UiPath\[Studio_version]\

How to Build Custom Rules

In its off-the-shelf form, Workflow Analyzer comes with a wide set of rules.

However, if your team's needs are not covered by the pre-defined rules, you have the option of building custom rules.

To build custom rules, you need **the UiPath.Activities.Api** package from the Official feed.

Just like custom activities, **custom rules are built in an IDE**.

They can be integrated in Studio at global or project level.

Role of Governance

Governance plays an important role in ensuring Studio users adhere to development standards and comply with certain rules by enforcing policies.

One of the main features of Governance include to use Workflow Analyzer **rules to enforce policies**.

Example: You can determine which applications and URLs users are allowed to automate by configuring App/URL Restrictions rule in the Workflow Analyzer.

As a part of governance policies, organizations can enforce custom rules pertaining to their needs across all projects, in a centralized manner using one of the **two ways**:

UiPath Automation Ops	**File Based Governance**
A web-based application available in Automation Cloud, used to create and deploy governance policies.	Create a JSON policy file and deploy the file locally, externally, or through Orchestrator

Governance - Best practices

- The governance file must be of **type.config** with the following name **uipath.policies.config**

- The Deployment page overwrites any file-based governance you had previously in your environment when you visit it for the first time

- As it is expected that any file-based governance to no longer apply, make sure you import your already existing policies to Automation Ops and deploy them from here

- If Automation Ops is enabled for your organization, you are prompted to select whether to generate a Classic or a Modern policy
  - Select the classic option for the file-based governance model
  - The modern option generates policies that you can then import in Automation Ops for an easy migration to that model

Resources

Topic	Link
Governance - File-based Governance	https://docs.uipath.com/studio/standalone/2022.10/user-guide/governance
Governance - UiPath Automation Ops	https://docs.uipath.com/automation-ops/automation-cloud/latest/user-guide/governance
UiPath Studio Guide - Command-Line Support	https://docs.uipath.com/studio/standalone/2022.10/user-guide/about-workflow-analyzer
UiPath Studio Guide - Building Custom Rules	https://docs.uipath.com/studio/standalone/2022.10