Improvements to Error Messages

Checklist

☐	User Stories Documented
☐	User Stories Reviewed
☐	Design Reviewed
☐	APIs reviewed
☐	Release priorities assigned
☐	Test cases reviewed
☐	Blog post

Introduction

Contextual error messages provides meaningful information to CDAP users which can help them move forward in error situations. However, sometimes error messages can be vague and without context which can lead to an unpleasant user experience. Providing meaningful error messages to the users is a challenge. In many error situations, for the same type of error, the error message format is different. This inconsistency can confuse the users. Having a central repository of errors would help developers reuse the same error messages as well as formulate error messages in a consistent way.

When user is encountered with an error situation, error message can point users to appropriate actions. Sometimes the remedy would require users to follow a certain steps. For example, in wrangler, when a deployed database driver does not contain the driver class, users would need to repackage the jar and redeploy the jar. In this case, having a catalog of errors would help users to quickly find the steps to follow to fix the underlying issue. One approach to solve this challenge is to surface error codes to users along with error messages. Error codes would help users search the catalog with an error code and find the remedial steps.

Goals

There are three goals which needs to be achieved to improve error handling:

- Provide a guideline to write better error messages: General best practices on formulating a meaningful Error Message
- Add a framework to add error codes to the error messages so that users can browse the catalog to figure out remedial steps.
- Add a framework to standardized error messages and put them at central location for better visibility.

Scenarios

• Scenario 1: Errors in Wrangler

• Scenario 1.1

Alice wants to wrangle data using CDAP's Wrangler tool. As part of that, Alice wants to connect to Database Source using Wrangler Connection. While attempting to do that, Alice is seeing a cryptic error message while testing the connection that she does not know how to resolve just by looking at the error message. In order figure out cause of the issue and recommended action, Alice will like to browse error code catalog with the error code displayed along with the error message.

• Scenario 1.2

Alice is applying transformations to the data on the fly using CDAP's Wrangler tool. However, while applying transformations, Alice sees an error message that does not suggest recommended action to fix the issue. She wants to browse the error code catalog to figure out recommended actions to resolve the issue.

- ### Scenario 1.3

  Alice wants to wrangle data using CDAP's Wrangler tool. While applying transformations to the connected source data on the fly, Alice tries to parse boolean column as csv. However, Alice observes that built-in directives to parse boolean data as csv and avro returns different error messages. Alice would like to see standard error message from both the directives when column type is incompatible.

- ### Scenario 1.4

  Alice wants to wrangle data using CDAP's Wrangler tool. While connecting to the source Alice uses user name and password stored in secure store. However, the secure store keys are not resolved. Alice wants to look at the error code and share it with the CDAP developer to figure out the underlying root cause.

- ## Scenario 2: Errors in Pipeline

  - ### Scenario 2.1

    Alice is a pipeline developer who has built a pipeline using CDAP data pipeline studio. While running the pipeline, it failed with no meaningful error message in logs.To debug this issue further, Alice wants to share error code from the logs with Bob, the data pipeline app developer to figure out what is the cause of the error.

  - ### Scenario 2.2

    As a data pipeline developer, Alice is trying to get schema from the database source. While attempting to do that from CDAP data pipeline studio, Alice sees an error message. In order to figure out the cause and action for the error message, Alice will like to search the error code in the error catalog.

  - ### Scenario 2.3

    As a data pipeline developer, Alice is trying to validate the pipeline. While attempting to validate it, Alice saw an error message that requires fixing the plugin property. Alice want to have an error code along with the error message so that she can search the probable cause on error catalog.

  - ### Scenario 2.4

    As a data pipeline developer, Alice is using secure macros in the pipeline, however, the secure macros are not resolved. Alice wants to use the error code and share it with CDAP developer to figure out the root cause.

- ## Scenario 3: Errors in CDAP

  - ### Scenario 3.1

    As a CDAP user, Alice wants to upload a plugin to CDAP. However, Alice is not deploy a plugin and gets an error. Alice wants to use the error code to browse the catalog to figure out the cause and recommended action.

  - ### Scenario 3.2

    As a CDAP user, Alice wants to enable Wrangler to wrangle data. However, Wrangler service is not starting up.Alice wants to use the error code so that it can be shared with the support engineer for further debugging.

  - ### Scenario 3.3

    As a CDAP user, Alice wants to enable Wrangler to wrangle data. However, Wrangler service is not starting up.Alice wants to use the error code so that it can be shared with CDAP developer to figure out the root cause.

Design Discussions

```java
/**
 * Represents error codes.
 */
@Beta
public enum ErrorCode {
  // 11 prefix for directive errors
  UNSUPPORTED_ENCODING_TYPE("WRA-111001"),

  // 12 prefix for handler errors
  DRIVER_CLASS_NOT_FOUND("WRA-121001"),
  INVALID_MACRO_ERROR("WRA-121102");
  .....


  // 13 prefix for storage
  ....

  private String code;

  ErrorCode(String code) {
    this.code = code;
  }

  /**
   * Returns the code for this error.
   */
  public String getCode() {
    return this.code;
  }
}
```

ErrorInforProvider.java

```java
/**
 * Error information provider.
 */
@Beta
public interface ErrorInfoProvider {

  /**
   * Returns error code associated with the error.
   * NOTE: this interface returns special error code for backwards
compatibility.
   */
  default ErrorCode getErrorCode() {
    return ErrorCode.UNDEFINED_CODE;
  }

  /**
   * Returns an array of arguments to formulate error messages.
   */
```

```
    default Object[] getErrorArgs() {
      return null;
    }
  }
```

Look up in the exception vs highest level (Handler and Wrangler Transform)

- should be done in the exception - What if for some reason resolution fails?

DirectiveLoadException.java
```
/**
 * Class description here.
 */
public class DirectiveLoadException extends Exception implements
ErrorInfoProvider {
  private static final MessageLookup LOOKUP = new MessageLookup();
  private ErrorCode errorCode;
  private Object[] errorArgs;

  ...

  public DirectiveLoadException(ErrorCode errorCode, Object... args) {
    super(LOOKUP.get(errorCode.name(), args));
    this.errorCode = errorCode;
    this.errorArgs = args;
  }

  @Override
  public ErrorCode getErrorCode() {
    return errorCode;
  }

  @Override
  public Object[] getErrorArgs() {
    return errorArgs;
  }
}
```

Impact on UI

Test Scenarios

Test ID	Test Description	Expected Results

Bug Fixes

- **CDAP-14378** - Getting issue details... STATUS

  CDAP-15040 - Getting issue details... STATUS
- **CDAP-15499** - Getting issue details... STATUS
- **CDAP-15593** - Getting issue details... STATUS
- **CDAP-14797** - Getting issue details... STATUS
- **CDAP-15426** - Getting issue details... STATUS
- **CDAP-15581** - Getting issue details... STATUS

Releases

Release 6.1.0

Future work