



CONTAINERCON @



THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
NORTH AMERICA

Standardizing Errors: A Practical Guide with Dapr

Cassie Coyle

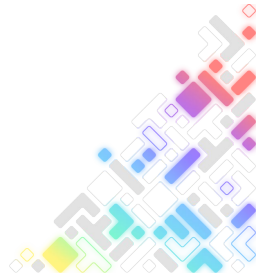


#osummit



Table of Contents

- Introduction to Dapr
- Importance of Error Standardization
- Richer Error Model
- Errors pkg in dapr/kit repo
- Status of Error Standardization in Dapr
- Conclusion



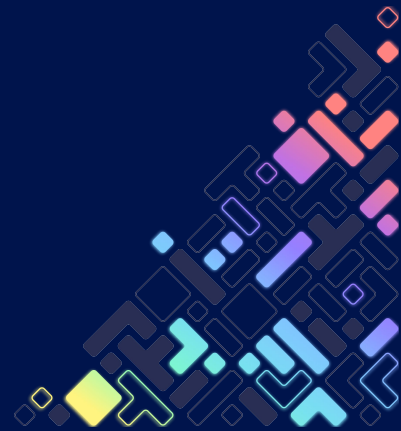
Speaker & Team



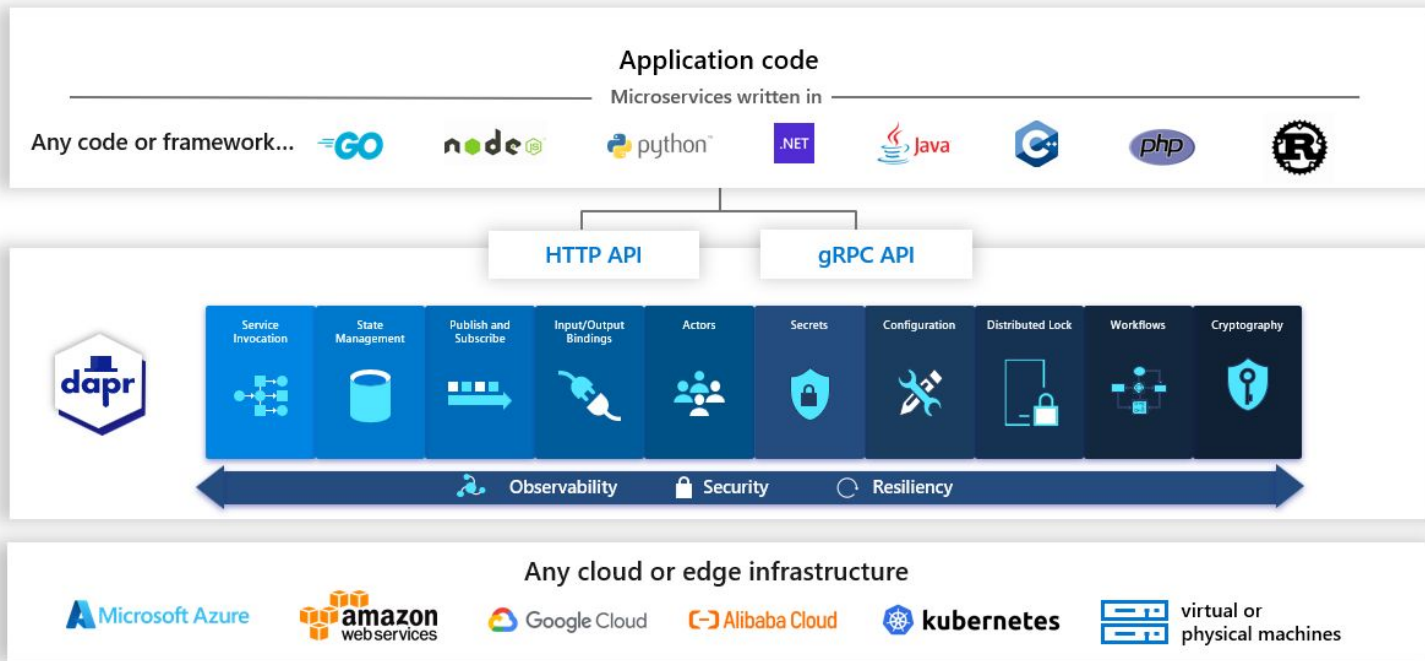
Cassie Coyle
Software Engineer



Introduction



Dapr



Dapr Community Feedback

*Frankly, our whole team loves Dapr. This is the
backbone of our services.*

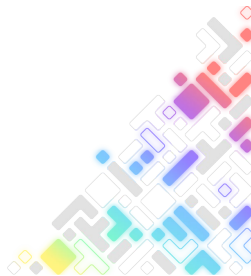
DEVELOPER, LARGE SERVICES CO.

“

*Dapr doesn't have a lot of expense at the
beginning, and there is a lot of payback.*

DEVOPS ENGR. LARGE FINANCIAL SERVICES CO.

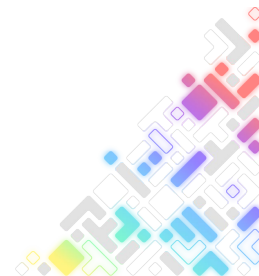
“



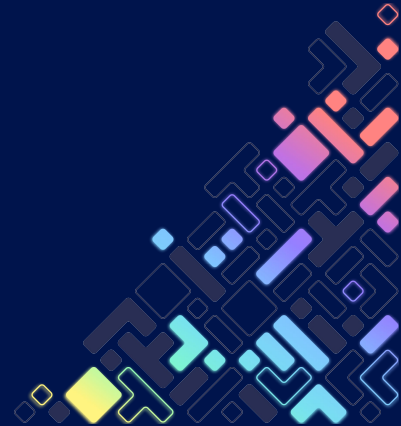
Dapr Community Feedback

What do you dislike about Dapr?

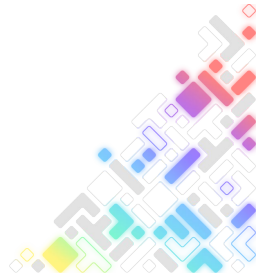
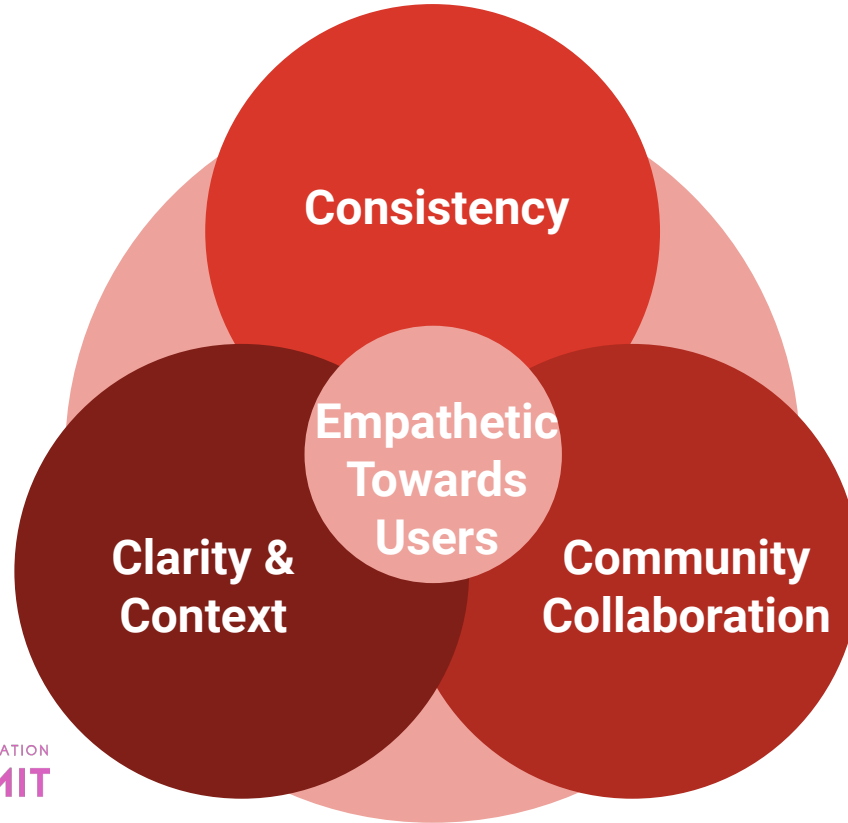
Debugging/troubleshooting is hard



Importance of Error Standardization



The Importance of Error Standardization



Richer Error Model



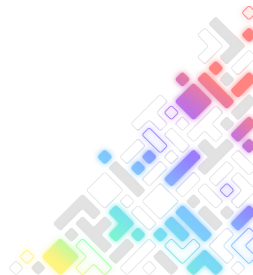
Richer Error Model

```
package google.rpc;

// The `Status` type defines a logical error model that is suitable for
// different programming environments, including REST APIs and RPC APIs.
message Status {
    // A simple error code that can be easily handled by the client. The
    // actual error code is defined by `google.rpc.Code`.
    int32 code = 1;

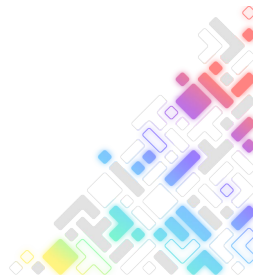
    // A developer-facing human-readable error message in English. It should
    // both explain the error and offer an actionable resolution to it.
    string message = 2;

    // Additional error information that the client code can use to handle
    // the error, such as retry info or a help link.
    repeated google.protobuf.Any details = 3;
}
```



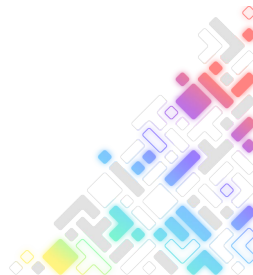
Error Details

- `ErrorInfo` (required)
- `RetryInfo`
- `DebugInfo`
- `QuotaFailure`
- `PreconditionFailure`
- `BadRequest`
- `RequestInfo`
- `ResourceInfo`
- `Help`
- `LocalizedMessage`
- `QuotaFailure_Violation`
- `PreconditionFailure_Violation`
- `BadRequest_FieldViolation`
- `Help_Link`



Error Detail

- **ErrorInfo** (required)



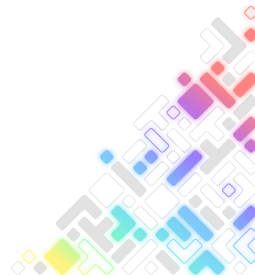
Error Detail

- **ErrorInfo** (required)

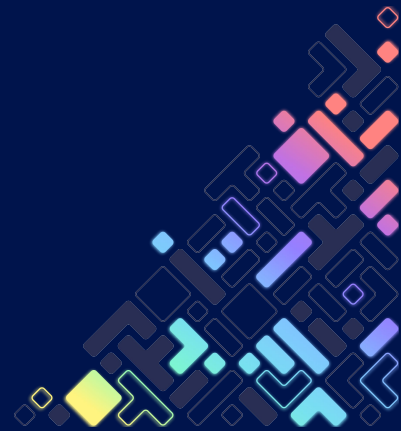


ErrorInfo:

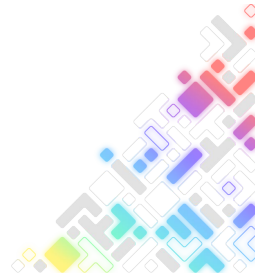
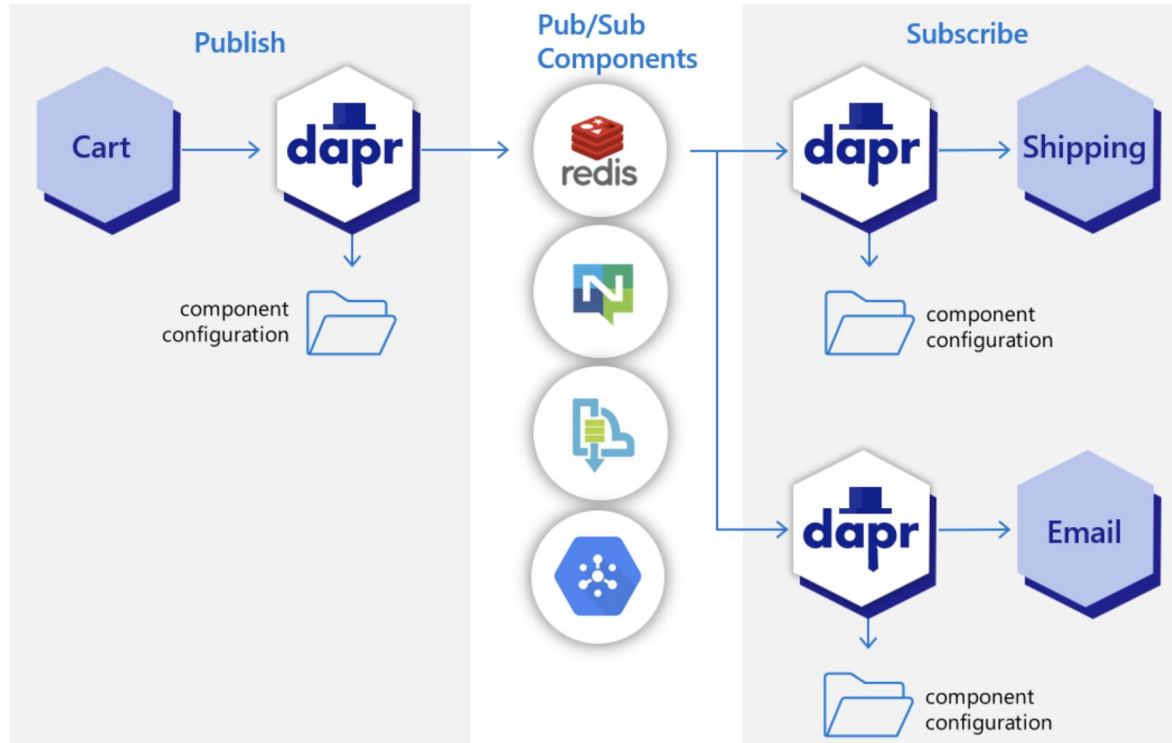
- Domain: dapr.io
- Reason: <reason>
- Metadata: <metadata>



Enriched Errors Before & After



PubSub API Overview



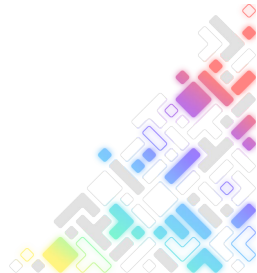
Before Enriched Error for PubSub API

```
$ grpcurl -H 'dapr-app-id: sub' -d '{"pubsub_name":"fakeKafka",  
"topic":"fakeTopic"}' -plaintext localhost:57534  
dapr.proto.runtime.v1.Dapr.PublishEvent
```

ERROR:

Code: InvalidArgument

Message: pubsub **fakeKafka not found**



After Enriched Error for PubSub API

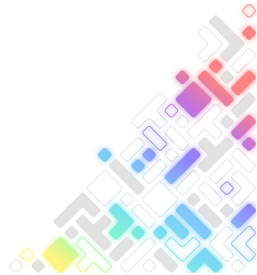
ERROR:

Code: InvalidArgument

Message: pubsub fakeKafka is not found

Details:

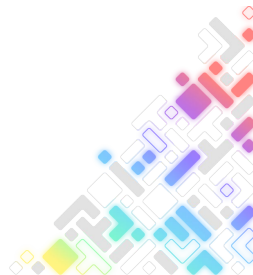
```
1) {  
  "@type": "type.googleapis.com/google.rpc.ErrorInfo",  
  "domain": "dapr.io",  
  "reason": "DAPR_PUBSUB_NOT_FOUND",  
}  
2) {  
  "@type": "type.googleapis.com/google.rpc.ResourceInfo",  
  "description": "pubsub fakeKafka is not found",  
  "resourceName": "fakeKafka",  
  "resourceType": "pubsub"  
}
```



Expanded Enrichment

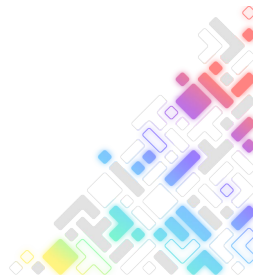
Details:

```
1) {  
  "@type": "type.googleapis.com/google.rpc.HelpLink",  
  "url":  
    "https://docs.dapr.io/developing-applications/building-blocks/pubsub/pubsub-overview",  
  "description": "Dapr docs pubsub overview",  
}
```

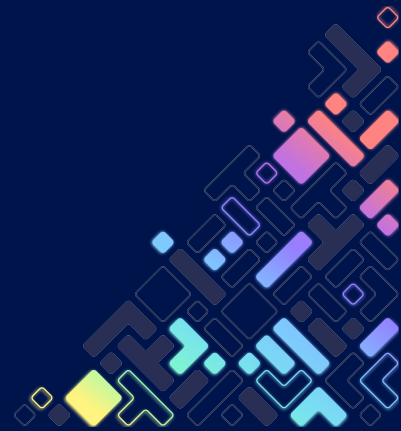


Error Details

- `ErrorInfo` (required)
- `RetryInfo`
- `DebugInfo`
- `QuotaFailure`
- `PreconditionFailure`
- `BadRequest`
- `RequestInfo`
- `ResourceInfo`
- `Help`
- `LocalizedMessage`
- `QuotaFailure_Violation`
- `PreconditionFailure_Violation`
- `BadRequest_FieldViolation`
- `Help_Link`

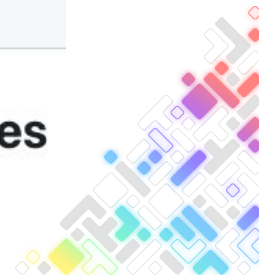
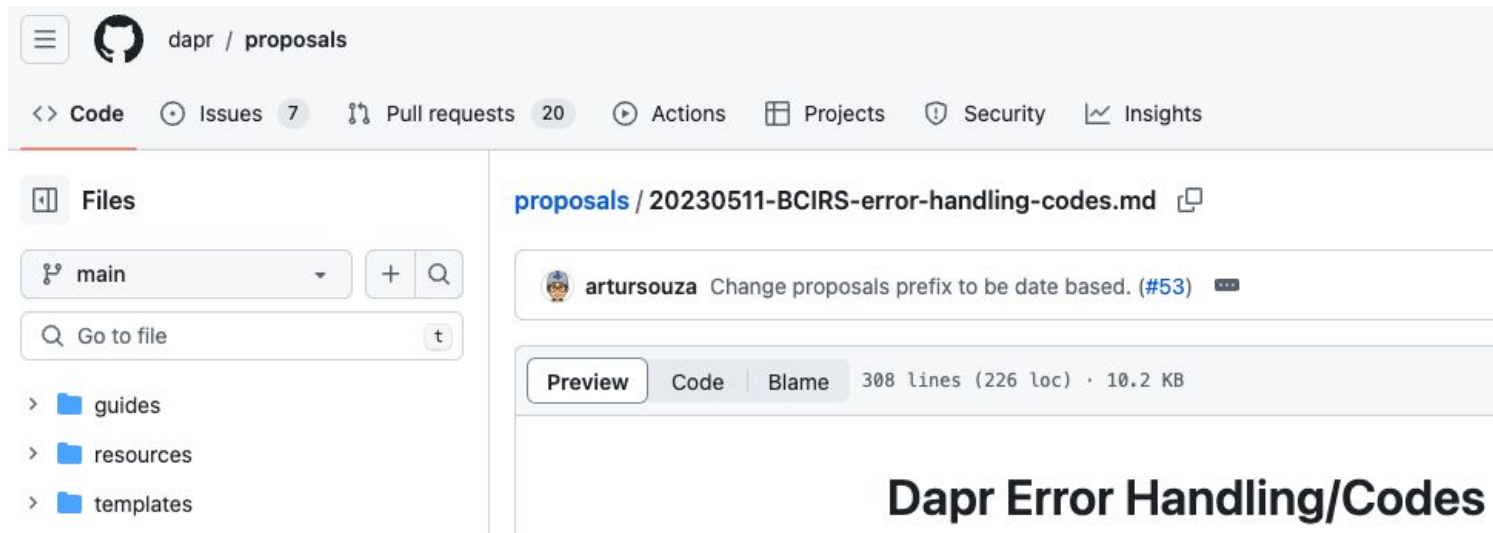


Errors pkg in the dapr/kit repo

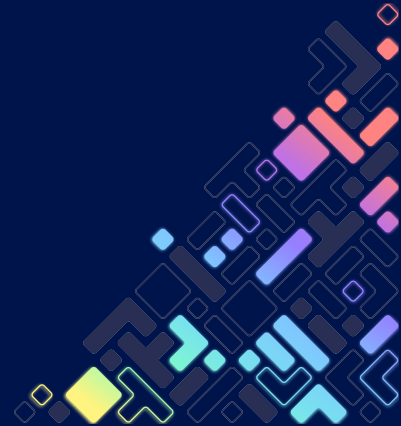


Errors pkg in the dapr/kit repo

- Builder Pattern
- Error immutable
- Backwards compatibility
- Based off the dapr/proposals



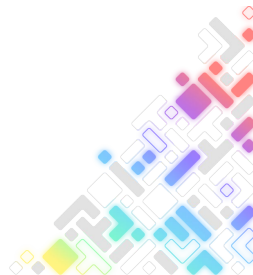
Consumption of the Errors pkg



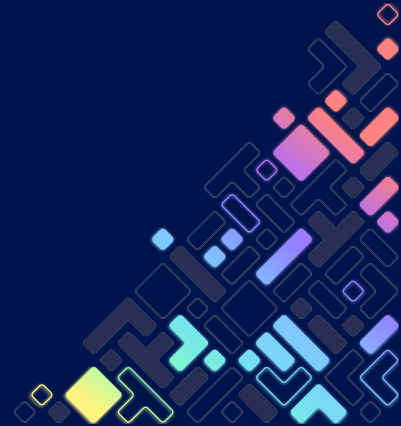
Consumption of the Errors pkg

```
err := errors.NewBuilder(grpcCode, httpCode, msg, tag)
```

```
return err.WithErrorInfo(  
    errors.CodePrefixPubSub+errCode,  
    p.metadata,  
).Build()
```



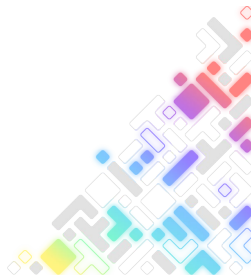
Implementation Details



Implementation Details of the Errors pkg

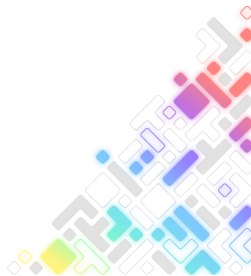
```
type Error struct {  
    details []proto.Message  
    grpcCode grpcCodes.Code  
    httpCode int  
    message string  
    tag string  
}
```

```
type ErrorBuilder struct {  
    err Error  
}
```



Implementation Details of the Errors pkg

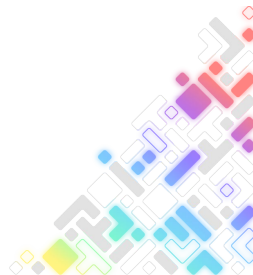
```
func NewBuilder(grpcCode grpcCodes.Code, httpCode int, message string, tag
string) *ErrorBuilder {
    return &ErrorBuilder{
        err: Error{
            details: make([]proto.Message, 0),
            grpcCode: grpcCode,
            httpCode: httpCode,
            message: message,
            tag: tag,
        },
    }
}
```



Implementation Details of the Errors pkg

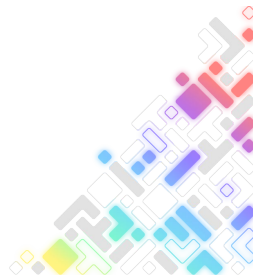
```
func (b *ErrorBuilder) WithErrorInfo(reason string, metadata map[string]string)
*ErrorBuilder {
    errorInfo := &errdetails.ErrorInfo{
        Domain:  Domain,
        Reason:  reason,
        Metadata: metadata,
    }
    b.err.details = append(b.err.details, errorInfo)

    return b
}
```



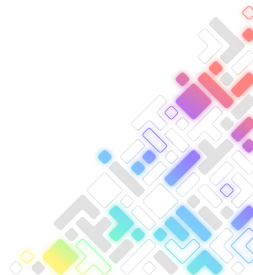
Implementation Details of the Errors pkg

```
func (b *ErrorBuilder) Build() error {  
    // Check for required ErrorInfo  
    containsErrorInfo := false  
    for _, detail := range b.err.details {  
        if _, ok := detail.(*errdetails.ErrorInfo); ok {  
            containsErrorInfo = true  
            break  
        }  
    }  
    if !containsErrorInfo {...}  
  
    return b.err
```

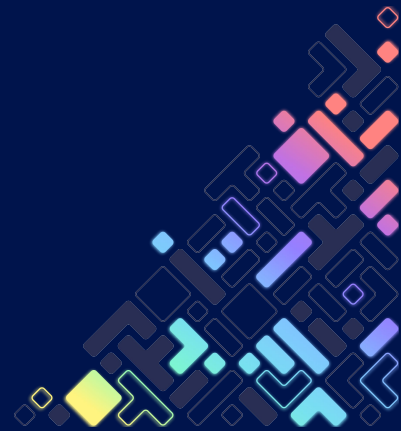


Implementation Details of the Errors pkg

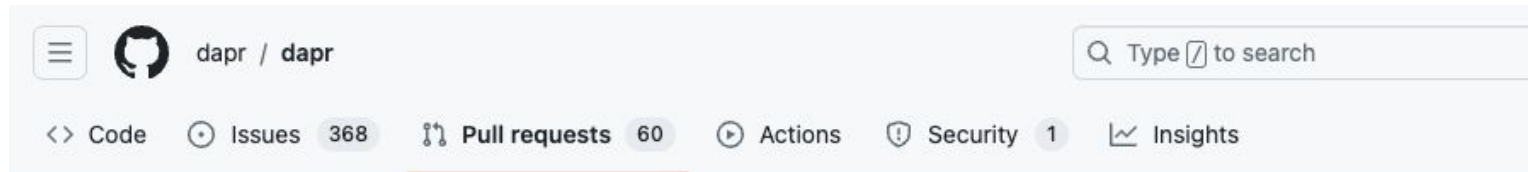
```
func (b *ErrorBuilder) Build() error {  
    // Check for required ErrorInfo  
    containsErrorInfo := false  
    for _, detail := range b.err.details {  
        if _, ok := detail.(*errdetails.ErrorInfo); ok {  
            containsErrorInfo = true  
            break  
        }  
    }  
    if !containsErrorInfo {...}  
  
    return b.err
```



Status of Error Standardization in Dapr



Enriched APIs



Richer error codes model #7257 **State API**

Merged

PubSub API: standardize err msgs #7322 **PubSub API**

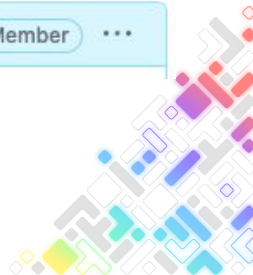
Conversation

Merged yaron2 merged 60 commits into `dapr:master` from `cicoyle:feat-standardize-err-msgs` on Jan 16

eler

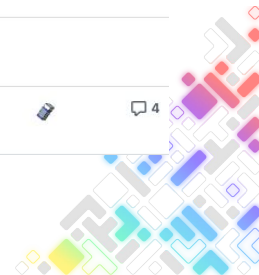
Conversation 53 Commits 60 Checks 22 Files changed 22

cicoyle commented on Dec 19, 2023 • edited Member

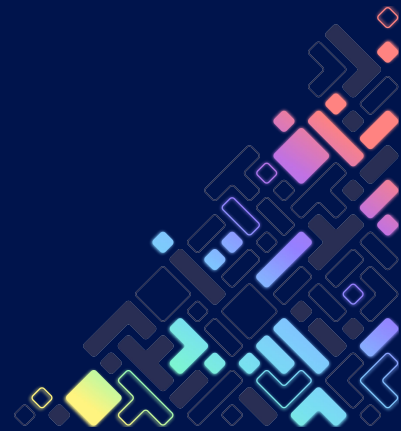


Efforts in Dapr

14 Open ✓ 3 Closed		Author	Label	Projects	Milestones	Assignee	Sort
🕒	Error Standardization: Cryptography API	good first issue	#7491 opened on Feb 2 by cicoyale	📄 5 tasks	🔗 1	👤	💬 2
🕒	Error Standardization: Placement API	good first issue	#7490 opened on Feb 2 by cicoyale	📄 5 tasks		👤	💬 2
🕒	Error Standardization: Metadata API	good first issue	#7489 opened on Feb 2 by cicoyale	📄 5 tasks			💬 3
🕒	Error Standardization: Health API	good first issue	#7488 opened on Feb 2 by cicoyale	📄 5 tasks			
🕒	Error Standardization: Workflow API	good first issue	#7487 opened on Feb 2 by cicoyale	📄 5 tasks			💬 1
🕒	Error Standardization: Distributed Lock API	good first issue	#7486 opened on Feb 2 by cicoyale	📄 5 tasks		👤	💬 1
🕒	Error Standardization: Configuration API	good first issue	#7485 opened on Feb 2 by cicoyale	📄 5 tasks			
🕒	Error Standardization: Secrets API	good first issue	#7484 opened on Feb 2 by cicoyale	📄 5 tasks			
🕒	Error Standardization: Bindings API	good first issue	#7483 opened on Feb 2 by cicoyale	📄 5 tasks			
🕒	Error Standardization: PubSub API	good first issue	#7482 opened on Feb 2 by cicoyale	📄 5 tasks			
🕒	Error Standardization: State API	good first issue	#7481 opened on Feb 2 by cicoyale	📄 5 tasks			
🕒	Error Standardization: Service Invocation API	good first issue	#7480 opened on Feb 2 by cicoyale	📄 5 tasks			
🕒	Error Standardization: Actors API	good first issue	#7425 opened on Jan 22 by cicoyale	📄 5 tasks		👤	💬 4



Conclusion



Conclusion

- Introduction to Dapr
- Importance of Error Standardization
- Richer Error Model
- Errors pkg in dapr/kit repo
- Status of Error Standardization in Dapr

