Four images that will not load

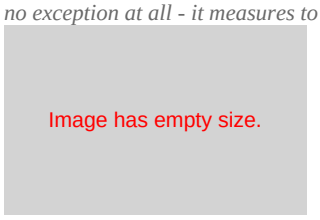
Each of the four below is an IImageSource written to fail, and each fails at a different point of the render. PinataLayout draws a placeholder where the picture would have gone and carries on - which is the contract, because one unreadable image should not cost a five hundred page report - and raises an event saying what happened. The next page is that event, collected.

A type nothing can decode

throws while XImage is built, before any measuring

An image of no extent

no exception at all - it measures to nothing

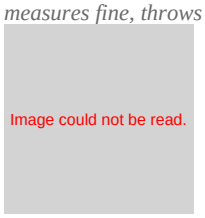


A file that ends too soon

throws while being measured

A stream that dies on the way out

measures fine, throws while being drawn



What the handler was told

4 failures, each reported once, at the moment its placeholder was drawn. The exception is the instance that was thrown - not a message, not a copy - so a handler can log it, rethrow it, or match on its type.

Image.Name	Failure	Exception	Message
unsupported.xyz	InvalidType	InvalidOperationException	xyz is not an image format anyone knows.
empty.png	EmptySize	none	no exception was thrown
truncated.png	NotRead	InvalidDataException	The file ends in the middle of the header.
unreadable.png	NotRead	IOException	The stream was closed by the other end.

Why an event and not a throw. PinataLayout's contract is that a document with a bad image still renders, and callers depend on it. Throwing would be the simpler change and the wrong one. The event leaves the contract alone while making the reason reachable, which is what issue 366 asked for - the exception used to go to Debug.WriteLine and nowhere else, which a release build compiles away entirely.

The fifth kind. ImageFailure has five values and only four appear above. FileNotFound has a placeholder string of its own but nothing in this fork ever assigns it: images arrive through IImageSource rather than by path, so there is no file for the renderer to fail to find. It is left in the enum because removing a public value would break callers switching on it.

Measured or drawn. The last of the four is the one worth remembering. It measures perfectly and fails only when its bytes are asked for, by which time the layout has been decided around an image that is never going to arrive - so the placeholder is exactly the size the picture would have been, and the page does not reflow. The other three are caught while measuring, and their placeholder is sized by SetFallbackDimensions instead.

See docs/specs/image-failure-reporting.md for what was wrong before this and why each part of it is the way it is.