



Random Fact 11.1

The Ariane Rocket Incident

The European Space Agency (ESA), Europe's counterpart to NASA, had developed a rocket model called Ariane that it had successfully used several times to launch satellites and scientific experiments into space. However, when a new version, the Ariane 5, was launched on June 4, 1996, from ESA's launch site in Kourou, French Guiana, the rocket veered off course about 40 seconds after liftoff. Flying at an angle of more than 20 degrees, rather than straight up, exerted such an aerodynamic force that the boosters separated, which triggered the automatic self-destruction mechanism. The rocket blew itself up.

The ultimate cause of this accident was an unhandled exception! The rocket contained two identical devices (called inertial reference systems) that processed flight data from



The Explosion of the Ariane Rocket

measuring devices and turned the data into information about the rocket position. The onboard computer used the position information for controlling the boosters. The same inertial reference systems and computer software had worked fine on the Ariane 4.

However, due to design changes to the rocket, one of the sensors measured a larger acceleration force than had been encountered in the Ariane 4. That value, expressed as a floating-point value, was stored in a 16-bit integer (like a short variable in Java). Unlike Java, the Ada language, used for the device software, generates an exception if a floating-point number is too large to be converted to an integer. Unfortunately, the programmers of the device had decided that this situation would never happen and didn't provide an exception handler.

When the overflow did happen, the exception was triggered and, because there was no handler, the device shut itself off. The onboard computer sensed the failure and switched over to the backup device. However, that device had shut itself off for exactly the same reason, something that the designers of the rocket had not expected. They figured that the devices might fail for mechanical reasons, and the chances of two devices having the same mechanical failure was considered remote. At that point, the rocket was without reliable position information and went off course.

Perhaps it would have been better if the software hadn't been so thorough? If it had ignored the overflow, the device wouldn't have been shut off. It would have computed bad data. But then the device would have reported wrong position data, which could have been just as fatal. Instead, a correct implementation should have caught overflow exceptions and come up with some strategy to recompute the flight data. Clearly, giving up was not a reasonable option in this context.

The advantage of the exception-handling mechanism is that it makes these issues explicit to programmers—something to think about when you curse the Java compiler for complaining about uncaught exceptions.
