What is the purpose of the tool "serialver"? How is it used?

When an object is serialized, its fully qualified class name, as well as the 64-bit SUID, is written to the stream. Later, when a class attempts to read the serialized object, it is important that its SUID matches that of the serialized object, as otherwise an InvalidClassException is thrown.

But therein lies the problem - because, when classes evolve, so do their SUIDs. But luckily, there is a solution at hand in the form of serialver - a utility provided by the Java SDK, which allows us to generate the SUID of a Java class file. If the output of serialver is included within a class file, then that class is now compatible with all it's persisted objects and can be used to read back the same, even though the class definition itself may undergo some mutation (e.g.addition/removal/update of methods) down the road.

For instance, running:

```
serialver foo.Person
```

generates:

```
foo.Person: static final long serialVersionUID =3734178553292263588L;
```

Now, you can continue to read the foo.Person objects even with future versions of the class, as long as they have the "old" SUID embedded within them as:

```
static final long serialVersionUID =3734178553292263588L;
```

This will work because a class, before it reads the serialized data from the stream, first checks for the presence of the serialVersionUID field within. If found, its value is written to the stream, instead of one being calculated on the fly. Since the SUID values should now match, there would be no problem in reading in the serialized data.