0:00:00.560,0:00:06.000
So today the floor is open to NIST team, to David

0:00:06.000,0:00:08.240
Waltermire, OSCAL technical director

0:00:09.120,0:00:16.400
and Dmitry Cousin, team member. They
are going to go over some of the NIST

0:00:16.400,0:00:28.480
open source tooling. So, David, I
think that you are the first... right? Thank you, Michaela. I'm going
to share my slides while I'm doing that.

0:00:29.200,0:00:32.640
I just wanted to say welcome everyone and thank

0:00:32.640,0:00:37.600
you for the opportunity to present
some of the work we've been doing

0:00:38.480,0:00:41.880
here at NIST. So we've been working

0:00:41.880,0:00:52.080
to see if I can close this door. So in this,
we've been working on a variety of different types

0:00:52.080,0:01:01.840
of tooling to support our work on the
team around developing OSCAL. But

0:01:01.840,0:01:05.360
we've been also keeping an eye towards building

0:01:06.560,0:01:11.840
open source software that we
believe will be broadly useful

0:01:11.840,0:01:20.360
to the larger community. And so we've been
building the software with that thinking in mind

0:01:20.360,0:01:26.880
. So today I'm going to talk a little bit about
some of the software that we currently have

0:01:26.880,0:01:39.840
available, as well as some software that we're

currently working on for a future release.

0:01:44.080,0:01:46.720
So before I get into specific tools,

0:01:46.720,0:01:51.040
I wanted to talk a little bit about the
types of tooling that we're working to

0:01:51.040,0:01:53.040
provide here at NIST

0:01:54.120,0:01:58.560
. Some of what we provide are
what we would call programming

0:01:58.560,0:02:04.000
language APIs. The intent there is to support

0:02:04.960,0:02:08.560
development of, you know,
business logic functionality

0:02:09.680,0:02:14.320
on top of our skill based
formats as quickly as possible.

0:02:15.760,0:02:24.720
So the idea is you can use a library which will
give you, you know, basic parsing capabilities

0:02:24.720,0:02:30.480
to marshal OSCAL content into
native objects, which you can then

0:02:31.840,0:02:35.040
manipulate and build higher level application

0:02:35.040,0:02:40.640
on top of. So our our thoughts there is that,
you know, these types of programming APIs

0:02:41.440,0:02:44.560
provide an easier on ramp for developers who are

0:02:45.520,0:02:52.080
trying to implement tools around scale and will
ultimately support a greater amount of innovation.

0:02:53.840,0:02:56.320

We're also working
on tooling that supports

0:02:56.880,0:03:00.720
content conversion. So OSCAL right now has

0:03:03.520,0:03:08.880
seven current models eight total
planned models and point 1.1 release

0:03:09.600,0:03:12.640
at the moment. And those models are

0:03:12.640,0:03:20.240
available in three different formats. So at
times three, that's twenty four different

0:03:22.480,0:03:29.440
format model specific formats that that we
that we currently maintain. We recognize that

0:03:30.320,0:03:31.840
different parties are going to prefer

0:03:32.400,0:03:36.960
the use of different, different
formats. I may prefer YAML for editing

0:03:36.960,0:03:42.240
and maybe JSON for tooling or
XML. We want to make it very

0:03:42.240,0:03:49.920
easy to to take a piece of OSCAL content
against any model in any format and convert it

0:03:50.560,0:03:52.240
to an appropriate format.

0:03:52.240,0:03:59.840
for use. And this type of content conversion helps
to provide a greater adaptability and flexibility

0:04:01.360,0:04:02.240
within

0:04:02.240,0:04:08.560
the use of OSCAL content. And then
finally, we're working on content editing

0:04:08.560,0:04:12.400
capabilities, specifically
browser based application

0:04:12.400,0:04:16.480
that support visualizing and editing

0:04:16.480,0:04:22.400
various OSCAL content. And our focus
there is providing tooling and a browser

0:04:22.400,0:04:28.560
that has a minimal footprint that doesn't require
that you install anything or get approval to

0:04:29.840,0:04:32.080
install anything. We want to make it easy

0:04:32.720,0:04:35.040
for for all of you to use

0:04:37.560,0:04:45.760
. So one tool that I've been working on for a
few years now is, well, it's actually a library

0:04:45.760,0:04:47.520
called liboscal-java

0:04:49.080,0:04:58.080
. So this this library is a Java
API library. It provides basic

0:05:01.120,0:05:08.640
code to to help you pars OSCAL data and
any of the available formats that allows you

0:05:08.640,0:05:13.280
then to operate on that data.
And in some specific ways,

0:05:14.400,0:05:18.240
this is an open source
project that is published out

0:05:18.240,0:05:24.400
on GitHub. You can follow the
link here. Access the code

0:05:25.400,0:05:29.120
. So liboscal-Java has a variety of features.

0:05:30.400,0:05:36.400
One feature that it supports is what
we call constraint validation. So

0:05:36.400,0:05:41.280
OSCAL as some of you might know,
that OSCAL is actually modeled

0:05:41.280,0:05:46.080
using a technology that we built called
metaschema. It's basically a way that we

0:05:46.080,0:05:55.680
can represent the structure of a model in a
format agnostic way. Many schema allows us to to

0:05:55.680,0:05:58.560
define constraints so we can

0:05:59.200,0:06:04.640
we can say things like if you provide this
value, then you must provide this other value or

0:06:05.680,0:06:14.880
we can say things like, you know, these like
three three fields, you must use one of the

0:06:14.880,0:06:20.080
three. These types of constraint
checks are not very well supported

0:06:20.080,0:06:24.320
and in common schema technologies like Excel

0:06:24.320,0:06:30.080
and JSON schema that we've developed. The
constraint system that allows us to express

0:06:30.080,0:06:33.840
express a richer set of constraints
on top of the OSCAL models

0:06:33.840,0:06:40.000
. So this liboscal-Java
tool supports enforcing those constraints

0:06:40.000,0:06:46.160
on the content that it passes.
It's also a multi format parser,

0:06:46.160,0:06:53.760
so it's capable of reading any of the
OSCAL formats. XML, JSON and YAML

0:06:53.760,0:07:01.920
, and then writing out content ... to any of
those those formats so you can read content and

0:07:01.920,0:07:05.200
... and write it back out.
in and YAML

0:07:06.360,0:07:16.560
. It does that by marshalling the data to sort
of a common object model that is independent of

0:07:16.560,0:07:21.120
the underlying format. And then
basically writing from that independent

0:07:21.680,0:07:24.880
object model back out to the selected format

0:07:25.480,0:07:28.880
, it provides builders using

0:07:29.440,0:07:36.160
the builder pattern, which allows you to
very easily construct some of the common

0:07:36.160,0:07:42.720
objects that exist within OSCAL. Things
like props and links and controls and

0:07:44.320,0:07:45.920
responsible parties and

0:07:46.720,0:07:52.000
and those types of constructs. They all have
builders which really simplify and make easy

0:07:53.200,0:07:55.760
generating content for these

0:07:55.760,0:08:05.440
types of objects. And it also provides a profile
resolver that's currently experimental, but

0:08:05.440,0:08:09.840

it is thanks to all of the great
community feedback we've been working to

0:08:11.040,0:08:14.880
make it better and better. And this supports

0:08:14.880,0:08:20.160
taking an OSCAL profile in any
of the formats and performs

0:08:20.160,0:08:23.840
the profile resolution operation on that profile.

0:08:24.560,0:08:33.680
Generating resulting resolved catalog. That's all
based on the latest specification that's out on

0:08:34.400,0:08:40.320
the OSCAL website.

0:08:40.320,0:08:47.680
So using the liboscal-Java, we're also
producing Java based command line tool

0:08:48.960,0:08:50.720
called oscal-cli

0:08:52.600,0:09:01.200
. So oscal-cli basically exposes all of the
features that liboscal-java supports

0:09:02.480,0:09:11.280
using an easy to use command line interface.
It makes it easy to convert between content by

0:09:12.320,0:09:16.880
loading that content and storing
it in an alternate content.

0:09:16.880,0:09:21.600
This allows you to write content and whatever
your favorite format is, and then convert

0:09:21.600,0:09:28.880
that content very easily for use in any of the
other formats, such as to import into a tool or

0:09:30.720,0:09:31.680
for publication.

0:09:33.120,0:09:36.560
It also allows you to quickly
validate ask out content using

0:09:36.560,0:09:40.880
the OSCAL constraints and
resolving profiles using

0:09:41.520,0:09:43.680
the profile resolver and

0:09:46.960,0:09:55.120
liboscal-java. I'd like to give you just a
brief demo of what this tool this tool functions

0:09:59.360,0:10:04.400
created a ... sheet here of
some commands to share with you. So

0:10:06.480,0:10:10.827
for this purpose, I have ...

0:10:10.827,0:10:12.640
. I have

0:10:15.920,0:10:17.840
downloaded

0:10:20.720,0:10:27.440
the OSCAL SP 800-53 Rev5 catalogue and
low baseline profile , both in XML

0:10:27.960,0:10:31.840
. I'll be using those to show you

0:10:32.400,0:10:37.360
both the profile resolution functionality of
the tool, as well as its ability to convert

0:10:38.920,0:10:45.840
it to convert content into alternate formats. So

0:10:46.880,0:10:52.880
First Command, I'm going to show you
this is the command that you would use to

0:10:53.680,0:11:00.560
convert and ask our catalog
from any format into a specific format.

0:11:01.600,0:11:05.600
The oscal-cli tool actually does
a deep inspection of the content

0:11:05.600,0:11:10.960
to determine what its its source format is.

0:11:10.960,0:11:15.440
And then so you don't have to actually
tell the tool what what that format is.

0:11:15.440,0:11:18.240
You do have to tell it what
format you want to convert to.

0:11:19.280,0:11:24.800
It can take to file arguments or a single
argument and when to file. Arguments are provided

0:11:24.800,0:11:29.120
. The first one indicates the
source content. The second one,

0:11:29.120,0:11:34.560
the destination file. If a single argument is
provided, it reads the source content. That's

0:11:35.120,0:11:38.080
that's specified and then writes
the output, the standard out.

0:11:39.120,0:11:44.240
So this is going to write a converted
OSCAL catalog and the YAML format to

0:11:44.240,0:11:45.120
the standard
out

0:11:46.840,0:11:50.400
. So it's reading the content end and then

0:11:51.680,0:11:56.800
processing of that content and is
writing it back out. You can see the

0:11:57.360,0:12:05.120
entire OSCAL catalog now in the YAML. I
can pipe this to my file if I want to.

0:12:06.400,0:12:09.840
So I
can call it like catalog dot

0:12:11.080,0:12:14.320
write

0:12:21.680,0:12:24.320
the next thing I wanted to show you is

0:12:27.040,0:12:28.640
the ability of the tool to

0:12:30.320,0:12:35.200
do profile a resolution. And one of the
interesting things about this tool is you can mix

0:12:35.200,0:12:39.600
, you can mix formats. So this is the command to

0:12:40.160,0:12:44.800
take a profile and resolve it. The
two argument indicates that you want

0:12:45.360,0:12:50.880
to write the profile out, the
resolved catalog out and JSON,

0:12:50.880,0:12:54.720
you can see. The first argument is
the profile, which is currently in XML

0:12:54.720,0:12:59.520
. And the second argument is the name
of the file that we want to convert

0:13:00.640,0:13:06.720
the that we want to store
it. The resolved catalog that

0:13:08.400,0:13:17.680
I'll run this XML run, the profile resolver that I
already have in that file. So I need to specify the

0:13:19.360,0:13:21.120
overwrite command

0:13:28.000,0:13:33.840
after to that over with the

0:13:51.040,0:14:01.840
they hold on one second.
Let me just read that file

0:14:17.000,0:14:18.000
.

0:14:18.000,0:14:19.120
And if that's still not right,

0:14:23.760,0:14:30.840
OK, the now it's resolving the
profile and writing that file that

0:14:33.520,0:14:38.640
which it's done. So now we can show you the

0:14:40.400,0:14:43.840
solved profile

0:14:51.200,0:14:52.560
in JSON format

0:14:53.560,0:14:59.040
. So that's the first few
lines of the resolved profile

0:15:00.280,0:15:09.040
. And then I can take that that profile
and convert it and this command will

0:15:10.640,0:15:12.640
convert

0:15:15.440,0:15:25.840
file to XML. All right, delete that

0:15:27.160,0:15:32.000
. And now it generated the XML version

0:15:32.000,0:15:39.840
of of that resolved profile
. The beginning of

0:15:41.280,0:15:47.040
it converted from JSON to XML now.
So this shows you like how quick and

0:15:47.040,0:15:54.320
easy this tool will allow you to basically

move between between Pascal content format

0:15:56.760,0:16:00.080
. That concludes my short demo

0:16:02.480,0:16:04.960
before I move on. You know,

0:16:04.960,0:16:11.680
this is just the start of some capabilities
that we ultimately intend to include in

0:16:12.640,0:16:20.080
and in this tool. Over time, we'd like to continue
to add features to it. One of the next things that

0:16:20.080,0:16:24.400
we'd like to start looking at
is adding functionality to

0:16:25.200,0:16:31.040
to take off scale content and
generate HTML or PDF documents

0:16:31.600,0:16:37.360
out of the oscal content
using a customizable template

0:16:38.920,0:16:48.800
. And we've got some existing XSLT based
tooling that we're currently using to do

0:16:48.800,0:16:53.520
things like generate PDFs and Excel
spreadsheets and things like that

0:16:53.520,0:16:58.320
for the eight hundred fifty three project.
One of the things that we want to work

0:16:58.320,0:17:04.720
on next is is generalizing that so that we
can integrate that into some of this tooling

0:17:04.720,0:17:07.920
for for more general use. So that'll

0:17:07.920,0:17:10.000
be a feature that should hopefully be coming in

0:17:10.720,0:17:14.400
the coming months as we continue
to make progress on that

0:17:16.360,0:17:23.920
. So we also have a variety of other additional
tools that I don't have time to demo today,

0:17:24.960,0:17:26.640
but I'd like to touch
on them a little bit

0:17:28.040,0:17:32.560
. So one area of tooling that we've

0:17:32.560,0:17:38.160
invested pretty heavily in is around our
continuous integration, continuous deployment

0:17:38.160,0:17:45.840
environment, which we use and both the main
OSCAL repo as well as in the scale content repo.

0:17:46.880,0:17:47.760
And this kicked

0:17:47.760,0:17:54.080
the environment, supports
a variety of of operations

0:17:55.400,0:18:03.360
. It allows us to take the source matter
schemas for defining models and generate

0:18:05.600,0:18:08.640
XML and JSON schemas generate

0:18:10.000,0:18:14.440
the web based documentation that's
out on the pages and instead of sites

0:18:14.440,0:18:18.560
, it allows us to generate content converters,

0:18:19.920,0:18:24.880
XSLT based content converters
which can be used to to translate

0:18:24.880,0:18:28.240
content from external to

JSON or from JSON text. Now

0:18:29.680,0:18:34.720
all of this happens
automatically as we as we merge

0:18:34.720,0:18:43.120
pull requests into Developer Main, and this
type of automation saves us a ton of time.

0:18:44.480,0:18:49.920
We also have similar functionality that
exist out on the Ask Out content site,

0:18:51.040,0:18:53.520
which which we maintain

0:18:53.520,0:18:59.360
, which does things like automatically
convert content that we author in one

0:18:59.360,0:19:03.520
format to all of the other
corresponding formats. So

0:19:03.520,0:19:08.880
it allows us to ontheir content and JSON,
and then it will automatically generate

0:19:08.880,0:19:14.160
the XML and YAML variants or
if we write it in the next HTML

0:19:14.160,0:19:18.000
will automatically generate
the JSON and animal variants

0:19:18.000,0:19:22.080
as an example. And this also saves
us a bunch of time because we don't

0:19:22.080,0:19:26.120
have to manually convert
every file that we touch and

0:19:27.680,0:19:31.200
technically, the CD is capable of being run

0:19:31.200,0:19:36.240
in any repo with a little bit of setup, and

there is some instructions if you go out

0:19:37.760,0:19:42.560
at the U.S. desktop, OSCAL,
GitHub Repo and click on Build.

0:19:44.320,0:19:51.840
Another tool that we offer is
the OSCAL Deep Diff tool. So this tool

0:19:53.520,0:19:59.920
that is maintained by Nikita Wootten provides
context sensitive difference in capabilities.

0:20:01.600,0:20:04.720
So, you know, the idea
behind deep diff is that

0:20:06.160,0:20:09.680
if you're comparing a couple pieces of content

0:20:13.040,0:20:18.480
of our scale content, sometimes doing
just a simple difference over the document

0:20:18.480,0:20:19.840
is not sufficient

0:20:20.600,0:20:28.080
. Content may be moved from one section of the
document to another section of the document.

0:20:28.800,0:20:31.040
This was a problem that we

0:20:31.040,0:20:35.440
we ran into when comparing 800-

0:20:35.440,0:20:41.040
-53, rev 4 and rev5. Some
control content was merged with other controls

0:20:41.040,0:20:49.040
. Some was withdrawn. Some was created,
broken out as additional control enhancements.

0:20:49.040,0:20:51.200
And we had a really difficult time

0:20:51.200,0:20:58.400

sort of tracing the movements of content through
the document. So to try to provide a better way of

0:20:59.200,0:21:02.720
of understanding the
changes and that type

0:21:02.720,0:21:10.480
of underlying OSCCAL content, such as the
800-53 catalog, we built this deep death tool

0:21:10.480,0:21:13.360
. It allows you to effectively configure

0:21:14.160,0:21:20.640
how the tool does its comparison. So it allows
you to do things like ignore certain elements

0:21:20.640,0:21:23.040
that are relevant for comparison

0:21:23.040,0:21:30.400
or identify specific fields that should be
used more strongly when trying to identify

0:21:30.400,0:21:38.400
if to sub portions of a document and represent the
same concept. So you can do things like identify

0:21:40.640,0:21:43.840
and identify our field as a key,

0:21:46.000,0:21:50.120
as a key key correlate essentially,
and the difference in algorithm

0:21:50.120,0:21:55.200
. So if if this type of different thing
is something that you're interested in,

0:21:55.760,0:21:58.800
I would encourage you to go out
and look at the OSCAL deep diff

0:21:58.800,0:22:05.440
, the name is a little misleading. We we did
build it for use with our OSCAL, but the tool

0:22:05.440,0:22:11.560
itself has applicability really to any model

because of how you can configure the different

0:22:11.560,0:22:18.240
thing so it can be used with scale. But it can
also be used with any other type of JSON data

0:22:20.680,0:22:30.000
. Another area that we have quite a bit of tooling
around is in XSLT. If you go out to the pages

0:22:30.000,0:22:39.920
that nistgov slash ... usnistgov slash or scale dash
tool site, there's a variety of different XSLT

0:22:39.920,0:22:44.880
demos and utilities that
exist out there. A lot of them

0:22:44.880,0:22:48.880
are focusing around various
visualizations of different types of

0:22:48.880,0:22:54.160
scale content. This is something that
is being maintained by Wendell Pierce.

0:22:55.520,0:22:59.040
You'll also be able to find out there a

0:22:59.040,0:23:01.040
another tool that we call oscal-cat,

0:23:02.320,0:23:07.840
which we're working to to publish on
that site, which Dmitry will be demoing

0:23:08.960,0:23:17.440
and a quick moment. We also have a variety of
programming APIs that are currently in work

0:23:17.440,0:23:23.840
. So in addition to the Java API and ask
our Seelye tool that I showed you earlier,

0:23:24.640,0:23:27.760
we're also working on C-sharp

0:23:27.760,0:23:37.240
and TypeScript now announces APIs, as well as a
TypeScript notice based command line tool, as well

0:23:37.240,0:23:43.840
. These these are projects that are
kind of in their early inception

0:23:44.400,0:23:46.960
phase where, you know, building out some of

0:23:46.960,0:23:53.360
the like really basic capabilities right
now around, you know, loading in a model.

0:23:55.040,0:23:56.960
But the plan is to continue

0:23:56.960,0:24:03.920
to to work on these libraries to add more
and more features over time. And they're

0:24:05.040,0:24:06.960
following a similar architectural

0:24:06.960,0:24:14.080
style as the as the Java APIs is. So the
hope is over time, they'll be able to provide

0:24:14.640,0:24:20.720
similar kinds of features that the
the Java APIs currently provide. And

0:24:24.320,0:24:26.400
a number of members of our team

0:24:28.720,0:24:38.080
in the Nikita and Arminta are working in in these
areas, building out these types of APIs and

0:24:38.720,0:24:39.360
and tooling

0:24:41.880,0:24:45.840
. Before I hand off to Dmitry, I just wanted to

0:24:46.560,0:24:50.800
to let to let you know how you
can contribute to these efforts.

0:24:51.840,0:24:55.520
So like all of our skill, our Open-Source tooling

0:24:56.240,0:25:00.800
is intended to be a community
driven effort. So your

0:25:00.800,0:25:07.760
participation and these projects can directly
impact both the success of the project

0:25:07.760,0:25:10.880
as well as ourselves. Larger
success

0:25:12.800,0:25:16.560
There's a few ways that you can
use and contribute to these tools.

0:25:16.560,0:25:22.000
You can integrate support for our scale and your
tools using some of these open source projects

0:25:22.920,0:25:32.080
. Your feedback on the existing tooling
that we have is is absolutely critical. We

0:25:32.080,0:25:36.800
need to hear from you what you
think is working well with the tool,

0:25:36.800,0:25:41.760
what features you would like to see added. What

0:25:41.760,0:25:45.200
improvements you would like to
see versus how it currently works.

0:25:50.240,0:25:52.480
We also are interested in hearing from you

0:25:52.480,0:25:56.960
around what other types of tooling
you would like to see the US produce

0:25:56.960,0:26:02.880
. We don't have a huge amount of
bandwidth to build a lot of tooling, but

0:26:03.520,0:26:06.160
if you have good ideas on some commodity

0:26:06.160,0:26:10.800

, you know, tooling that could be
used by a wide swath of the community,

0:26:10.800,0:26:16.880
it might be something that we would be interested
in in building. And as always, you can contribute

0:26:16.880,0:26:21.680
to the development of any of these tools
since they're out there as open source.

0:26:21.680,0:26:26.880
But if you have ideas on a future that
you would like to add to one of these

0:26:26.880,0:26:31.920
tools, you can always contribute source
code and to implement that feature.

0:26:33.440,0:26:36.720
And if you're interested in coordinating with any

0:26:36.720,0:26:41.440
of the devs, you can reach
out to us to to do that. Um,

0:26:42.160,0:26:46.960
I wanted to highlight that we also
have an OSCAL tools page, which list

0:26:46.960,0:26:53.040
many of these tools, as well as some community
tools. The tools on that page are not

0:26:54.320,0:26:56.360
are not endorsed by by NIST

0:26:56.360,0:27:00.720
, but as a service to the community.
We are providing a listing there

0:27:01.680,0:27:05.840
so that you can see what is currently out there

0:27:05.840,0:27:22.480
. Um, I will reserve questions until the
very end. I'll let Dmitry go next . Dmitry are you ready to hand off
to you?
Yes. All right. I'll stop sharing

0:27:25.960,0:27:29.520
the below and meet recursion

0:27:31.040,0:27:47.840
and I'll try to share the screen in a second
. If I can find how to do it okay

0:27:51.240,0:27:53.280
Can you see my screen now?

0:27:58.200,0:27:59.200
.

0:27:59.200,0:28:04.800
Okay? Yes, Dmitry, we can see it. Awesome. So you

0:28:04.800,0:28:07.840
know, I'll be talking and present
an open security assessment language

0:28:08.560,0:28:15.110
catalog authoring tool,
which for short will call oscal-cat

0:28:15.110,0:28:23.200
I gave last presentation on unfinished version of
the same tool on May 18th this year, and OSCAL

0:28:24.480,0:28:30.960
CAT is supposed to do. It's supposed to take
a catalog model of one of the instance of

0:28:30.960,0:28:37.840
the catalogs. At the moment, it's only one
, and build allow users to build a profile.

0:28:38.640,0:28:43.360
And further idea was that
it's supposed to generate

0:28:43.360,0:28:49.120
what's called profile. And then marked as TBD
 to resolve profiles and catalogs.

0:28:49.120,0:28:52.480
So out of the whole hierarchy
of the OSCAL objects,

0:28:53.280,0:28:57.840
also it operates only OSCAL

catalog model and also profile models

0:29:00.600,0:29:08.880
. Currently, Cat does not have some things which
would make it considerably production ready

0:29:08.880,0:29:13.760
, but for professional users, it's
still usable. That doesn't have some

0:29:13.760,0:29:18.880
user perfection. It doesn't ask questions.
Are you sure you want to do this when you

0:29:18.880,0:29:23.947
can kill something critical? It doesn't
appear to work with multiple catalogs

0:29:23.947,0:29:26.800
since it is the work for the future,
because we have to figure out

0:29:26.800,0:29:30.240
how to work with multiple catalogs
and in terms of profiles properly

0:29:33.000,0:29:39.600
, it's not yet ready to add new controls,
which didn't exist in the profile,

0:29:39.600,0:29:41.840
but it has all the tooling necessary for

0:29:41.840,0:29:44.480
that's basically control. There's a control,

0:29:44.480,0:29:51.840
right? It's an object and typescript,
so it's not a problem and doesn't have

0:29:51.840,0:29:58.320
at the moment we remove .. it had a screen for back
matter editing, but very few people actually

0:29:58.320,0:30:04.320
want to deal with that yet. So it
doesn't have all this currently not

0:30:04.320,0:30:11.200
yet available features. So the tool

itself is made in ionic and angular

0:30:11.200,0:30:13.920
. If you're curious, versions are listed here.

0:30:15.440,0:30:23.501
Cat code mainly consists of
TypeScript, HTML, CSS, SAS

0:30:23.501,0:30:24.240
and JSON. it is also using

0:30:24.800,0:30:31.280
QuickTime. At the moment, QuickTime was
used as a library only and CSG Pipeline

0:30:31.920,0:30:37.920
to remove dependency from the code deployment.
But if you want local IT to generate types,

0:30:38.960,0:30:40.960
you might want to use the tool and

0:30:40.960,0:30:49.840
I saw Zach asking if there are TypeScript available
entities or catalogues for all OSCAL objects

0:30:50.960,0:30:58.080
like I'm using QuickTime also to generate every
single entity as interfaces for TypeScript,

0:30:58.080,0:31:00.560
and it allows you to generate the most TypeScript

0:31:01.440,0:31:10.480
interfaces every single object from the Schema
JSON schema and then allows you to re cast the

0:31:10.480,0:31:15.760
objects back in to JSON if you
want. So it's quite handy. So

0:31:17.040,0:31:20.240
you were asking. So I'm
recommending that there was also

0:31:20.240,0:31:25.440
a question about validation. Validation
at the human level for constant dynamic

0:31:25.440,0:31:29.840
relationships is not done by this
tool, either, but I'm using AGV

0:31:29.840,0:31:37.680
, the tools using AGV to validate downloaded entities and of the
latest version,

0:31:37.680,0:31:44.080
all the entities are actually pulled
out from GitHub storage of their

0:31:44.080,0:31:49.600
catalogs and GitHub storage
of the schemas and ... scripts.

0:31:49.600,0:31:59.680
Because I'm developing on a
Mac they are all based on Mac zsh, so that's basically the
whole thing and a unique, angular

0:31:59.680,0:32:04.880
will. Ionic is not the best tool
because it allows you to compile

0:32:04.880,0:32:08.720
the same source code for multiple
platforms. You can make desktop application

0:32:09.840,0:32:19.360
if you want using an electron compiler.
You can make iOS and Android application

0:32:19.360,0:32:25.440
if you feel like. But at the moment we
are targeting mainly the web application

0:32:26.240,0:32:27.840
and we rely upon

0:32:29.600,0:32:32.560
download the availability
internet, the availability

0:32:33.280,0:32:38.880
of the schema, files and
profile catalogs, baselines and

0:32:38.880,0:32:45.280
resolved baselines by internet. But in the

case, internet connection does not work.

0:32:46.320,0:32:48.800
Application hosts that are local copies

0:32:48.800,0:32:55.840
of those files and silently falls
back to the local files in internet

0:32:55.840,0:33:01.760
upload download was not successful
on every downloaded file. Also,

0:33:02.320,0:33:08.320
application run AGV validation to make sure
that catalogs pulled out over the internet

0:33:08.320,0:33:09.840
are actually valid. Catalogs

0:33:12.400,0:33:16.480
source will become open. I'm not sure
if the repository is already open or

0:33:16.480,0:33:20.960
will become open whenever we
consider the project is ready

0:33:20.960,0:33:27.040
. It's in the it used to be in May
18th when I gave the first presentation

0:33:27.040,0:33:30.960
I used two slides. It used to be
private repository. I'm not sure because

0:33:31.680,0:33:38.720
newsstand special procedure. So a few people,
including AJ and Nikita, were helping me to

0:33:41.760,0:33:49.680
make application publication
ready. So now I'll show if you

0:33:51.680,0:33:52.180
both

0:33:56.240,0:34:02.640
few screenshots because many of the screenshots,
luckily for the commendations, I had them

0:34:03.400,0:34:08.880
. So you start with the
tool or with authoring mode

0:34:08.880,0:34:13.040
and you have options to pick up
two catalogs which are available

0:34:13.040,0:34:16.400
at all GitHub region for revision five.

0:34:18.240,0:34:22.800
In this screen, you can see below
the big red frame two profiles

0:34:23.600,0:34:29.360
I was working on before and the goals in
the title what they were derived from,

0:34:29.920,0:34:33.840
which were agreed on over the council

0:34:35.120,0:34:42.240
brings the authoring mode also
has because it downloads the

0:34:42.240,0:34:49.120
catalogs after a certain timeout, which is
currently set up eight hours by default.

0:34:49.680,0:34:52.160
But they also edit settings because

0:34:52.160,0:34:56.480
I was annoyed that they couldn't test this
stuff unless I left the application

0:34:56.480,0:35:02.000
running for eight hours to shorten the
timeout. So when the catalog becomes

0:35:02.000,0:35:06.560
stale, the application warns you when you
select a particular type of the catalog

0:35:06.560,0:35:11.440
that your you did not refresh
the files from the internet

0:35:12.160,0:35:20.720
or from the local storage. So catalog
baselines and profile baseline

0:35:22.240,0:35:31.120
low, moderate and high for revision four or
low, moderate, high and privacy baselines also

0:35:31.120,0:35:34.880
can be refreshed using this interface

0:35:36.720,0:35:41.040
so that data information his
validation for the mandatory fields

0:35:42.000,0:35:49.760
for the if you don't feel democratized on
Goodreads and that's pretty much typical

0:35:50.800,0:35:58.080
life. So when you fill them in,
it shows the validation. So for

0:35:58.080,0:36:02.960
the field while you're in, if you're
out of the field or just turns regular

0:36:02.960,0:36:09.880
dark without special highlights.
So currently the rigorous controls

0:36:09.880,0:36:16.240
, because we're in process of making
drag and drop additions to the actual

0:36:18.480,0:36:23.040
are in work, but to select
controls, I edit and this is new

0:36:24.160,0:36:27.920
baseline, so it has options
chance of mark baselines, which

0:36:27.920,0:36:34.800
basically creates this highlights for sure
control belonging to a particular baseline. So

0:36:34.800,0:36:38.400
right now we're seeing five catalog and

0:36:38.400,0:36:44.640
baselines. Actually in this particular time
application, I was making screenshots from has some

0:36:47.040,0:36:56.080
circle or reference, so it does not pull
data properly from ... from the session. So

0:36:56.080,0:36:57.200
it's highlighted everything,

0:36:58.000,0:37:03.360
every single baseline present in this particular
catalog type, which is a revision five

0:37:04.360,0:37:09.840
. It also has an option to project baseline. So
basically when you have all the catalogs present,

0:37:10.480,0:37:15.280
all the controls present in a particular
version of the catalog, you have option to

0:37:15.280,0:37:20.560
recheck baseline and you can pick up
whichever baseline you want to recheck.

0:37:21.360,0:37:24.560
And after you click the
appropriate baseline button

0:37:25.360,0:37:30.880
basically pulls the check marks on every single
control, which belongs to the baseline and

0:37:32.960,0:37:36.160
also you have option to Taylor baseline,

0:37:36.160,0:37:42.480
which means you cut out only controls which
belong to the baseline and you end up with

0:37:42.480,0:37:49.440
the effectively resolved profile. But for PreCheck
Baseline, so I ordered slightly differently so

0:37:49.440,0:37:56.240
well, it was a little bit my fault. So
when you select, let's say, high baseline

0:37:58.160,0:38:01.760
and Project Baseline, it selects
controls, which belongs to

0:38:01.760,0:38:06.000
the baseline with the checkmark.
So basically it's allows you to

0:38:06.960,0:38:11.600
jump faster. Because I was put
personally, I was annoyed for a while

0:38:11.600,0:38:16.000
that I have to go and select things
with the hand when I was thinking,

0:38:16.000,0:38:19.760
OK, baseline is already handy
anyways, so why can't I do it?

0:38:22.000,0:38:25.840
And I decided to in the future.
So after you select the controls,

0:38:26.880,0:38:30.800
you operate on the subset of
the controls you selected and

0:38:30.800,0:38:39.600
I'll show you later on of the structures in the
profile entity. Public Master, the particular

0:38:40.880,0:38:45.840
subsections of the profile. So when you
select them, you can modify controls

0:38:49.680,0:38:53.360
using the modified buttons and after you're done.

0:38:53.360,0:38:57.040
I didn't show retro controls because
of the moment it's work in progress

0:38:57.680,0:38:59.840
and after you're done with the

0:39:02.160,0:39:06.320
all the tailoring you wanted to do,
the particular draft of the profile,

0:39:07.360,0:39:11.920
you have choices to either save
your workspace and progress and

0:39:11.920,0:39:17.840
it's allows you to make a snapshot of what you've
done or metadata of the selection of the controls

0:39:17.840,0:39:21.520
so that it controls basically all the work
in progress and drops. And then the session

0:39:21.520,0:39:25.840
. So later on, you can start
already from the place which you

0:39:26.560,0:39:31.760
already worked on. But
unfortunately, because it's to in the

0:39:31.760,0:39:38.160
application space, it's a web application. And
besides store and cookies, there is not much place

0:39:38.800,0:39:41.280
I can persist. Things because I like write

0:39:41.280,0:39:43.760
back to the server without
having actual web server

0:39:44.560,0:39:51.320
and apps or behind it. So there is
also option to save the whole work

0:39:51.320,0:39:57.360
space locally in the case somebody decides
to clean up browser space and so on.

0:39:57.360,0:40:01.040
You can at least would be
able in the future to upload

0:40:01.040,0:40:06.720
the file with the current state of your
editor things and hopefully resume your work

0:40:06.720,0:40:11.040
further. The right Red Square
highlighted is basically

0:40:11.040,0:40:15.760
allows you to save the actual
profile file on your disk locally,

0:40:15.760,0:40:20.960
or you can drop it still in up
space. Already compose profile so

0:40:20.960,0:40:25.840
I'll talk about the future features
we're planning. Oh, and the

0:40:27.120,0:40:30.480
well making presentation and
playing with the authoring mode

0:40:30.480,0:40:34.560
and statements of particular objects which are

0:40:35.360,0:40:40.400
downloaded from the internet, which
catalogs, profiles, baselines and so on.

0:40:40.400,0:40:45.520
I decided to add the settings screen.
It's literally app settings. They're

0:40:45.520,0:40:50.080
also leaving right now in the store. I
tried to save them and the cookies as

0:40:50.080,0:40:55.040
first books. Cookies are much more fragile
than the stores and the top with the store.

0:40:55.040,0:40:59.840
And they added this very scary button clear
state store. Because when you begin things

0:40:59.840,0:41:03.920
, sometimes they need to clean the state
store and then easier to have the button for

0:41:03.920,0:41:09.840
it. So I don't know. In the production,
I might consider removing it or I will

0:41:09.840,0:41:14.160
create some kind of level of access

and will allow it for power users,

0:41:14.160,0:41:19.600
but harder for non power users. I
don't know yet. It just then, using

0:41:19.600,0:41:23.760
as well, which I edited recently,
was scary. It worked out well

0:41:25.280,0:41:29.520
and right now in the application, I

0:41:29.520,0:41:32.960
saved the expiration interval and made it only

0:41:32.960,0:41:38.560
nine thousand of an hour, which is like a few
seconds. So original and vision features of the

0:41:41.440,0:41:46.000
was go cast were to do catalog,
which has a really simple structure

0:41:46.000,0:41:51.480
from the outside. Unless you go into the
details inside of the controls and groups of

0:41:51.480,0:41:56.720
, you should be able to feel
the profile. Very simple, right?

0:41:56.720,0:42:01.680
And then once you have profile, we were
thinking you'll use some future tools

0:42:01.680,0:42:06.640
which will allow you to pick up adjacent
objects, which OSCAL cut was the proof of

0:42:06.640,0:42:11.520
concept to show that OSCAL can be worked
on using pure JSON. But at the moment,

0:42:11.520,0:42:17.520
we don't know about any future JSON tools which
allow you to resolve profile expressed in JSON.

0:42:17.520,0:42:21.600
So it looks like we'll end

up taking JSON profile

0:42:21.600,0:42:28.160
them, convert it to XML, and using Saxon,
converting it to results and the result profile.

0:42:28.160,0:42:31.120
I saw somebody's question
also what is a result profile

0:42:31.120,0:42:34.720
. So basically, you build profile, which
tells the transformations you're applying

0:42:34.720,0:42:41.040
to your original catalog and profile
contains three essentially three important

0:42:41.040,0:42:44.800
parts. One, that's impulse, which tells
you which controls you want to pull in

0:42:45.520,0:42:50.720
or which controls you want to exclude. You
have Option two, well, let's say, include all

0:42:50.720,0:42:55.760
and then exclude particular one. So basically,
you're tailoring it. It's basically like

0:42:55.760,0:43:00.720
laying the application of the papers on
top of other paper, including something

0:43:00.720,0:43:05.440
during something the verbs in the
inside of imports input. Basically,

0:43:06.400,0:43:10.160
all this notion of including
something excluding something

0:43:11.168,0:43:15.520
numbers it, then it has modifications.
So when you modify controls,

0:43:15.520,0:43:20.480
modify parameters, modifications
list will contain modifications

0:43:20.480,0:43:25.120
you're doing, and then merge will describe
how the modifications will be applied to

0:43:25.120,0:43:30.480
the controls, which will include also
moves, the regrouping and so on. So

0:43:30.480,0:43:35.520
that was very simple. So planned features
and enhancements. Once we started developing

0:43:35.520,0:43:40.320
the simple total, there was a wish list
was growing. The moment they chose 18

0:43:40.960,0:43:45.600
items written on a piece of paper,
but the most commonly discussed ones.

0:43:46.400,0:43:50.000
So of course, I'm currently
we're going on fix bugs, which

0:43:50.000,0:43:54.400
were fine, try to make application
faster or at least make it feel

0:43:54.400,0:44:00.080
faster with the synchronous loads and
so on. Optimizing memory footprint

0:44:00.080,0:44:02.880
of the browser. Because sometimes I noticed that

0:44:02.880,0:44:08.640
when you work with the lot of objects and
immediately pull out all four baselines

0:44:09.760,0:44:13.360
resolved baselines which are pretty
big objects, catalogues and so on,

0:44:14.080,0:44:17.840
browser I have pretty beefed
up computer but the browser

0:44:17.840,0:44:22.560
sometimes makes my computer
sound like and from the future,

0:44:22.560,0:44:29.520
things were finding acceptable plug ins for
markdown editing and rendering below editing

0:44:29.520,0:44:38.800
of the pros for the controls to make it easier.
At the moment, the editing looks not pretty. Let's

0:44:38.800,0:44:44.000
put it this way for a group in controls,
we're planning to add drag and drop

0:44:44.000,0:44:48.480
to make it simpler at the moment, or
the lost working version had dropped

0:44:48.480,0:44:56.000
down. Since it's not as user friendly as one
would imagine, we'll want to a possibility to

0:44:56.720,0:45:00.000
start with an empty cat and
built from scratch everything. So

0:45:00.000,0:45:03.360
basically, you start with empty object
because interfaces are they are using

0:45:03.360,0:45:07.920
QuickTime. So it doesn't matter whether you
start with the full catalog or with an empty

0:45:07.920,0:45:12.160
one and just go. At the
moment, there is no UI to add

0:45:12.720,0:45:17.760
its own UI to select. So that
probably will result in development of

0:45:17.760,0:45:25.920
some other screens and UI. Plugins and resolution
of the profile probably will have to go through

0:45:25.920,0:45:31.236
the path of Saxon and then JSON
and XML, of the conversion

0:45:31.236,0:45:38.240

projects JSON of XML Resolved
Profile. And we add this discussing

0:45:38.240,0:45:40.240
this option of uploading.

0:45:41.600,0:45:47.560
So let's say you picked up catalog, you built
up your profile, you downloaded your profile

0:45:47.560,0:45:52.560
, but then you used external tooling or
in the future, you use our tool end to

0:45:52.560,0:45:57.520
resolve profile. So basically, you have
full catalog and then all you have right

0:45:57.520,0:46:01.440
now in UI is two original catalogs
or even five additional four. But

0:46:02.080,0:46:07.600
you have already catalog original profile,
which is a catalog you would like to use

0:46:07.600,0:46:10.240
at the moment because I'm downloading the

0:46:10.880,0:46:17.360
catalogs from online. It's not difficult
to let the user to actually upload

0:46:17.360,0:46:27.200
their own profile, result profiles, catalog
and start profile with it. It's just we have to

0:46:27.200,0:46:32.480
strictly verify the profile result profile
complies to the schema, and that's all

0:46:34.680,0:46:41.600
. So basically, all uploads of the profiles and
resolving profiles, validation and everything

0:46:41.600,0:46:43.920
should be able to done - that's another feature

0:46:44.760,0:46:48.880
. So if you have questions, we'll hold on to

0:46:50.280,0:46:54.480
the later. And right now I'll demonstrate
a little bit of the application.

0:46:55.040,0:47:00.480
So I left this application running
overnight so that our timeout because it now

0:47:01.360,0:47:06.800
and at the moment, you can see that I selected
it doesn't matter which one I select, I select,

0:47:07.360,0:47:10.160
let's say Revision 4 and it prompts me right

0:47:10.160,0:47:19.200
now immediately saying to refresh selected. So if
I select catalog baselines low, moderate and high

0:47:19.200,0:47:22.640
, so I refresh basically catalog object baselines

0:47:23.520,0:47:29.200
and all the baselines together. So this
checkbox basically opens the baseline

0:47:29.200,0:47:33.120
. So both of this particular type of
catalog when I refresh and stop scoping.

0:47:33.680,0:47:37.840
But I didn't refresh revision file. So again and

0:47:40.640,0:47:48.640
I can again select everything under
threshold. So I select the three and five go

0:47:48.640,0:47:55.200
ahead and it tells me here that I select
Revision 5 for the demo purposes I have

0:47:57.520,0:48:03.040
my timer for a demo purpose I have button
here, in the settings.

0:48:03.040,0:48:06.800
When I was showing you screenshots,
I had the check box for the demo, I

0:48:06.800,0:48:11.680
enables also two default roles, which are defined

0:48:13.120,0:48:16.720
inside of specification of
800-53, but it's not defined

0:48:16.720,0:48:23.040
in the Catalog. So I add this by default
and edit the few users. So I don't need to

0:48:23.840,0:48:26.560
actually type by hand and
say we found the presentation

0:48:26.560,0:48:33.520
. So I had the responsible party, let's say
font, and next to each other, you will see the

0:48:33.520,0:48:39.280
responsible party. So responsible parties show
up here and then notice for free. But still

0:48:39.280,0:48:46.160
you can add and so on. I will not edit the
rest of the things. So that's the longest

0:48:46.160,0:48:52.720
operation, which I will say, and we're
optimizing the time of the load, and I'm

0:48:55.200,0:48:57.520
that probably will be forced

0:48:59.440,0:49:08.080
to add some asynchronous things. So at the
moment, I didn't show before, I have some

0:49:09.600,0:49:11.920
this is developers branch. I have some

0:49:13.280,0:49:17.600
circular reference, which is
really annoying things to debug and

0:49:18.560,0:49:24.640
that's great when you have several
references for today's function of the

0:49:24.640,0:49:30.640

. But I made that stable just for the demo so
we can present. So when I say mark baselines

0:49:32.320,0:49:34.240
at the moment because session

0:49:36.160,0:49:39.200
has circle of reference of
marked all the baselines

0:49:39.200,0:49:44.240
which are in existence, for example, in control

0:49:44.240,0:49:49.760
is not here. I think I'm running out
of time. So this controls are not

0:49:49.760,0:49:54.480
completely mandatory training rides for
all the baseline, but they're all marked

0:49:54.480,0:49:59.440
because of the way the session is a circle
of reference and doesn't work properly.

0:49:59.440,0:50:04.240
Basically, all the same features as they
demonstrated before it with the static snapshots

0:50:05.520,0:50:13.520
have a possibility of checking baselines. I don't
think it will work at the moment because, yeah,

0:50:14.240,0:50:19.520
it does not because it's separate sessions back.
And so for reference there, but that's what I'm

0:50:19.520,0:50:25.840
debugging at the moment. I didn't do anything
. So let's say I'm selecting some small

0:50:27.880,0:50:32.160
. That's what I was talking about.
It becomes a really, really slow.

0:50:32.160,0:50:35.840
OK, so basically, I'm selecting controls

0:50:38.120,0:50:40.160
, trying to go to next

0:50:41.960,0:50:45.680
, and it didn't change

0:50:47.160,0:50:50.000
. It's just all right.

0:50:52.080,0:50:57.120
Basically, I'm kind of out of time, but in the
end, when you select, when you edit the controls,

0:50:57.760,0:51:00.480
you basically go to the download file

0:51:02.640,0:51:03.140
and

0:51:05.800,0:51:10.560
. That's what happens when you leave
application running for eight hours

0:51:13.280,0:51:21.920
in the screen. So just one last moment,
the JSON result profiles are coming from

0:51:21.920,0:51:23.200
historical school content

0:51:24.480,0:51:31.840
dynamically. Right now, the QuickType we're
using for resolving interfaces, we use schemas

0:51:31.840,0:51:36.720
for validating AGV and everything is pulled
in dynamically by the application. Over time,

0:51:36.720,0:51:41.200
we started and also gets repository is

0:51:41.200,0:51:45.440
using is the double spoke but then not
hundred percent sure if it's already

0:51:45.440,0:51:51.760
published from public or not, you can share
looking at all. And this was an illustration

0:51:51.760,0:51:57.280
to talk about profiles. I guess my time was
out. Well, finally, it got to the screen.

0:51:57.920,0:52:00.477
Sorry, the application becomes really
slow because I'm running from the bottom

0:52:00.477,0:52:06.400
up with everything. OK, so
if you have any questions

0:52:07.480,0:52:15.600
, please let me know. Here is the information
I'm talking. Thank you very much. And it is time

0:52:22.000,0:52:22.960
for questions now.