

Building Tools With The Learning Tools Interoperability Specification

MICHAEL D. HILBORN AND ARTIE BARRETT
NERCOMP ANNUAL CONFERENCE

MARCH 26, 2014

[HTTPS://GITHUB.COM/HARVARD-ATG/NERCOMP-LTI-BASIC](https://github.com/Harvard-ATG/NERCOMP-LTI-BASIC)

Agenda

- What is LTI?
- LTI tools in action
- How LTI works
- Writing an LTI tool
- Challenges to writing LTI
- Resources you can use
- Questions?
- Answers!



What Is LTI?

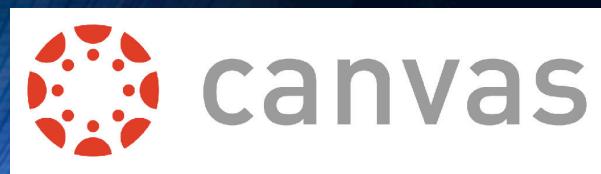
IN WHICH WE LEARN WHAT LTI IS AND PROPOSES TO DO



The Basics

*TLA: "Three Letter Acronym!"

- LTI is a TLA: "Learning Tools Interoperability"
- But first, let's consider another TLA, the LMS
 - (Learning Management System)
 - There is quite a landscape of LMS's



These LMS's offer a lot of tools, but not quite what I need. What if I want to write my own?

What if you wanted to write your own tool?

- For example, a tool that takes advantage of some of your research data
- You could write an independent tool and iFrame it
 - (But then no data can be passed from the LMS to the tool)
- Most LMS's offer ways of "plugging in" external applications
- This way your tool and the LMS can communicate!

So all I need to do is learn how my school's LMS extends itself, write my tool to its specs, and I'm done!

Not so fast, old sport. You are not thinking of the future!



Not so fast, indeed

- What if you wanted to share your tool with some colleagues at another University?
- But their University is running a different LMS?
- And what if your University decided to switch to another LMS?
 - (It's happened before!)

You see, my dear friend, writing for one LMS means rewriting your tool for another. A duplication of effort and resources!

I dare say, is there a solution?



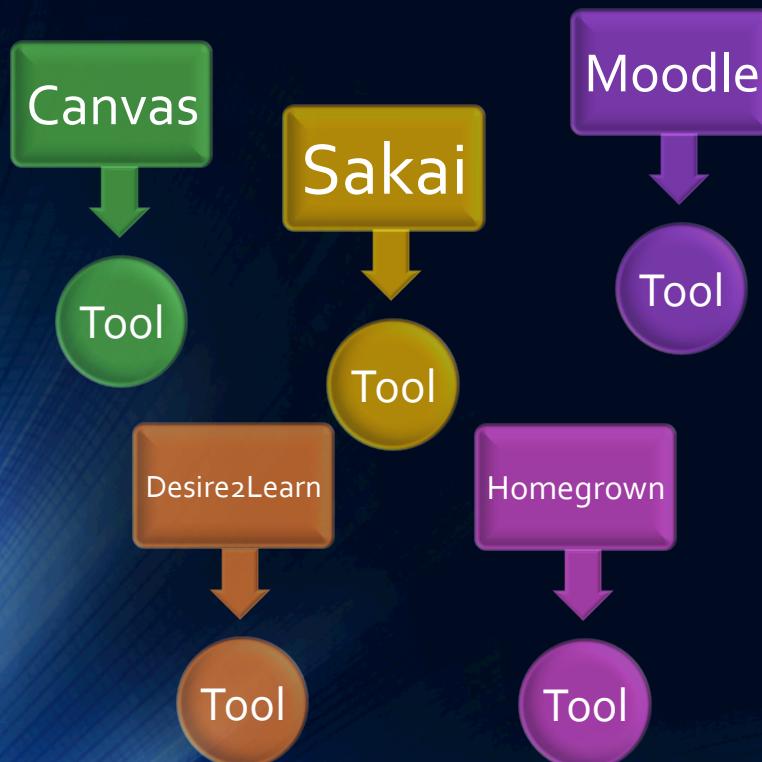
Yes, there is a solution... in the works...

- Welcome, Learning Tools Interoperability!
 - Initiated by the IMS Global Learning Consortium
<http://www.imsglobal.org>
- LTI attempts to:
 - Create standards that all LMS's and tools can implement so that they can speak with each other
 - Make your coding language of choice independent of the LMS
 - Make certain that when an LTI tool is implemented, it can work with **any** LMS
 - Do the above in a systematic, phased approach
 - (v1.0 -> v1.1 -> v1.1.1 -> v2.0)



So, LTI moves us forward...

FROM THIS...



TO THIS...



But the arrows are one-way! Shouldn't there be two-way communication?



Patience!
Get to
that we
will.

LTI Tools In Action

IN WHICH WE DEMONSTRATE A FEW LTI TOOLS INTEGRATED WITH CANVAS



How LTI Works

IN WHICH WE GO OVER THE BASIC CONCEPTS OF HOW LTI WORKS



Basic LTI (LTIv1.0)

- To start an external LTI tool, the LMS must first **launch** the tool
- A launch consists of two items sent to the tool:
 - An OAuth signature
 - A set of parameters (a few required, many recommended, others custom)
- In other words, a **signed HTTP POST request**
- Let's look at it in more detail...

Basic LTI Launch – What The User Experiences

A composite image illustrating the user experience of launching an LTI application. On the left, a young girl is shown from the waist up, sitting at a desk and looking intently at a laptop screen. A thought bubble originates from her head, containing the text: "I feel like learning about Dark Matter today. I'll just click here...". To the right of the girl is a screenshot of a web browser window displaying the Harvard Canvas interface. The URL in the address bar is <https://canvas.harvard.edu/courses/41/modules>. The browser's title bar shows "Course Modules: The Way". The Canvas navigation bar includes links for Apps, Harvard, Entertainment, Michael D. Hilborn, Inbox (3), Settings, Logout, and Help. The main content area is titled "Course Modules" and lists several items under "My Test LTI": "Learn LTI", "Post and Launch", "My Test LTI", "Hello World!", "LTI Sandbox: 8000", and "Hello Harvard". Below this is a section titled "Ted Talks" with items: "How many universes are there? - Chris Anderson" and "Dark matter: The matter we can't see - James Gillies". A blue curved arrow originates from the bottom of the girl's thought bubble and points to the "Click" button on a blue starburst graphic overlaid on the "Hello Harvard" item in the "My Test LTI" list. At the bottom of the browser window, a small orange bar contains the text "Thoughts on Canvas?".

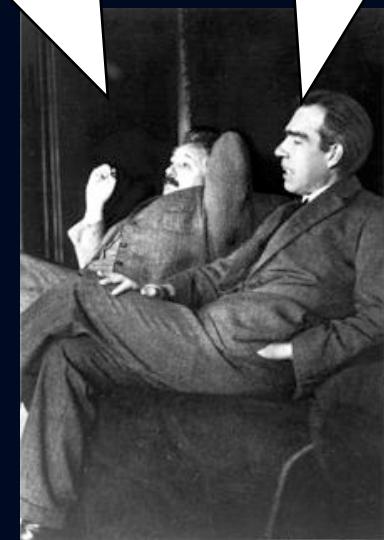
Basic LTI Launch – What The User Experiences



Oh, there's my content. That was relatively seamless...

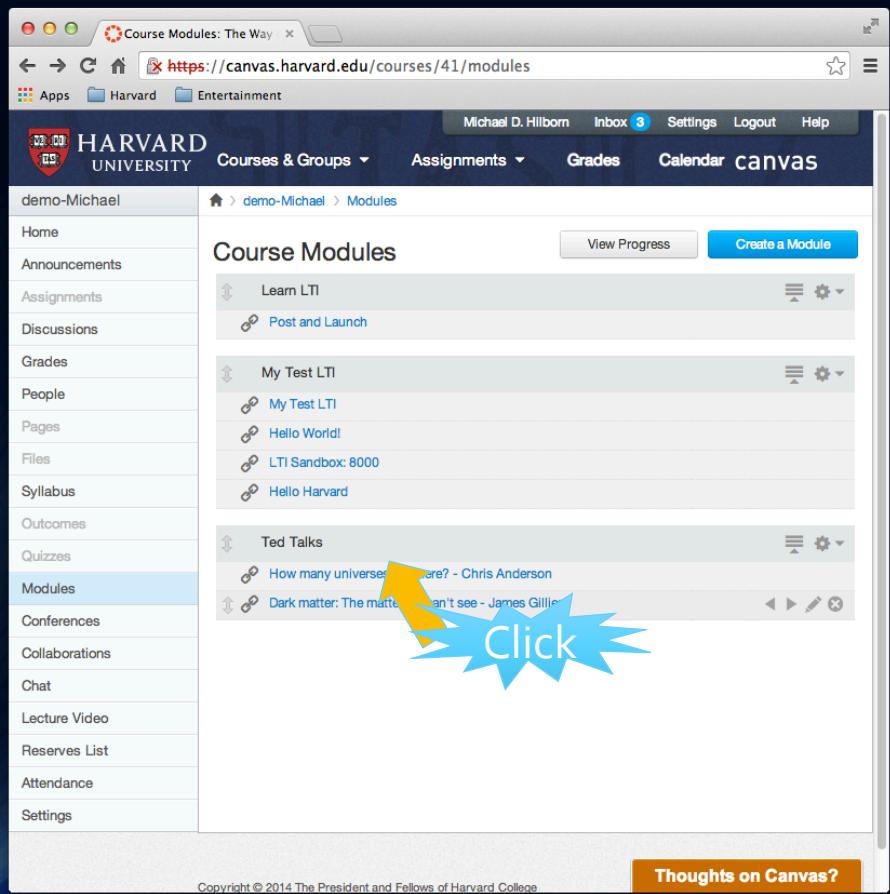
A screenshot of a web browser window showing a Harvard Canvas course page. The URL is https://canvas.harvard.edu/courses/41/modules/items/1512. The page displays a TED-Ed video titled "Dark matter: The matter we can't see - James Gillies". The video player shows a profile of a human head filled with various small images, the TED-Ed logo, and the text "Lessons Worth Sharing". The left sidebar of the Canvas interface is visible, showing links like Home, Announcements, Assignments, Discussions, Grades, People, Pages, Files, Syllabus, Outcomes, Quizzes, Modules (which is selected), Conferences, Collaborations, Chat, Lecture Video, Reserves List, Attendance, and Settings. A play button at the bottom of the video player indicates the video is currently playing at 0:12 / 5:34. A button labeled "Thoughts on Canvas?" is also visible.

Seems relatively simple enough!



Ah, but let's bore down into the mechanics of it.

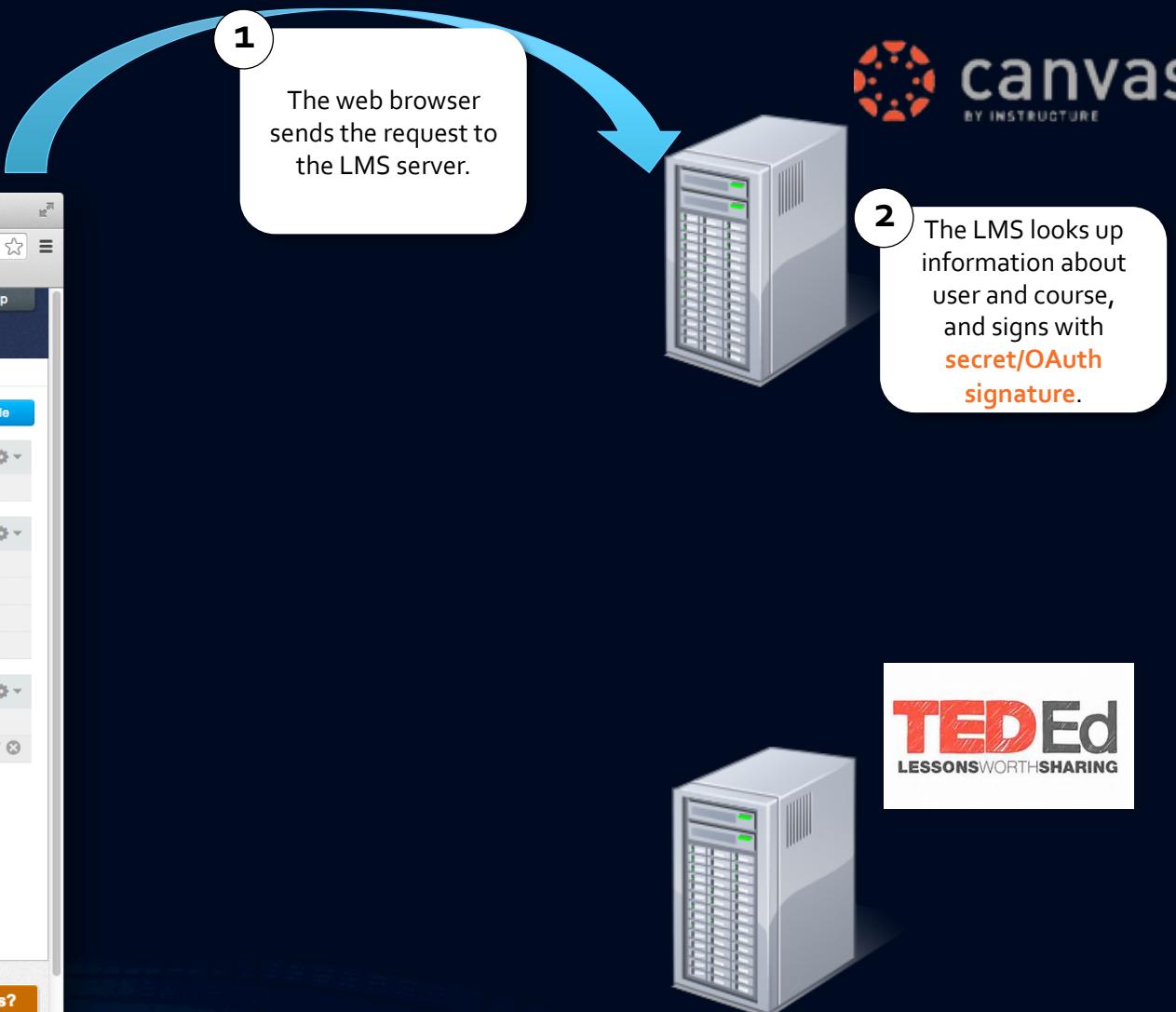
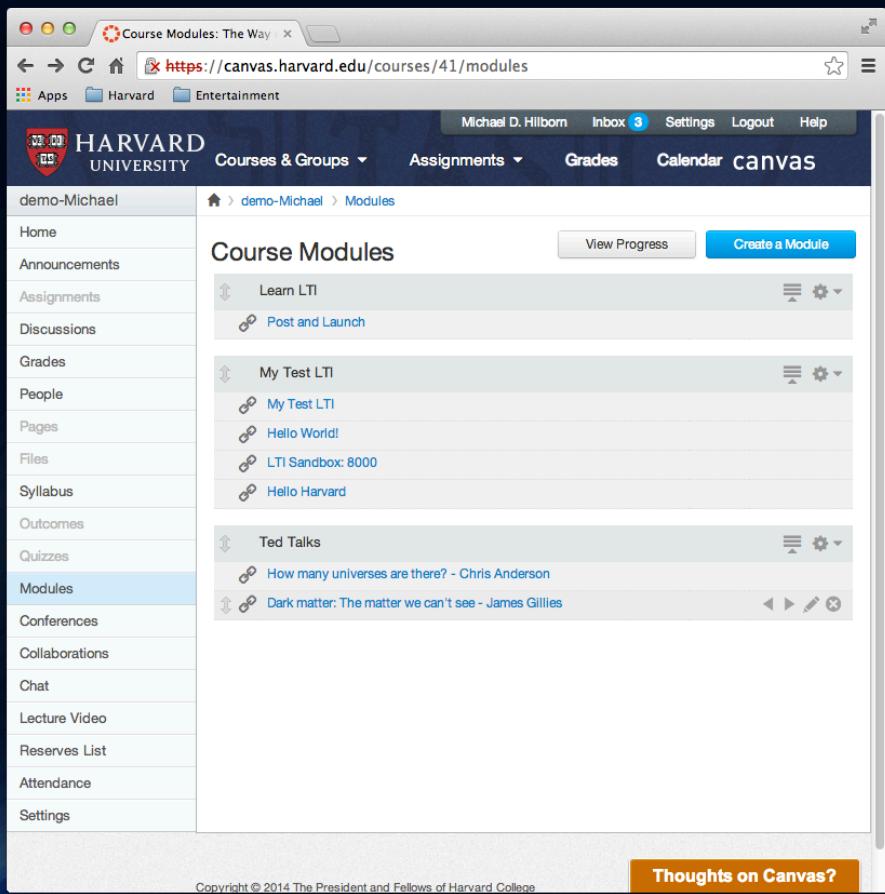
Basic LTI Launch – What The Web Experiences



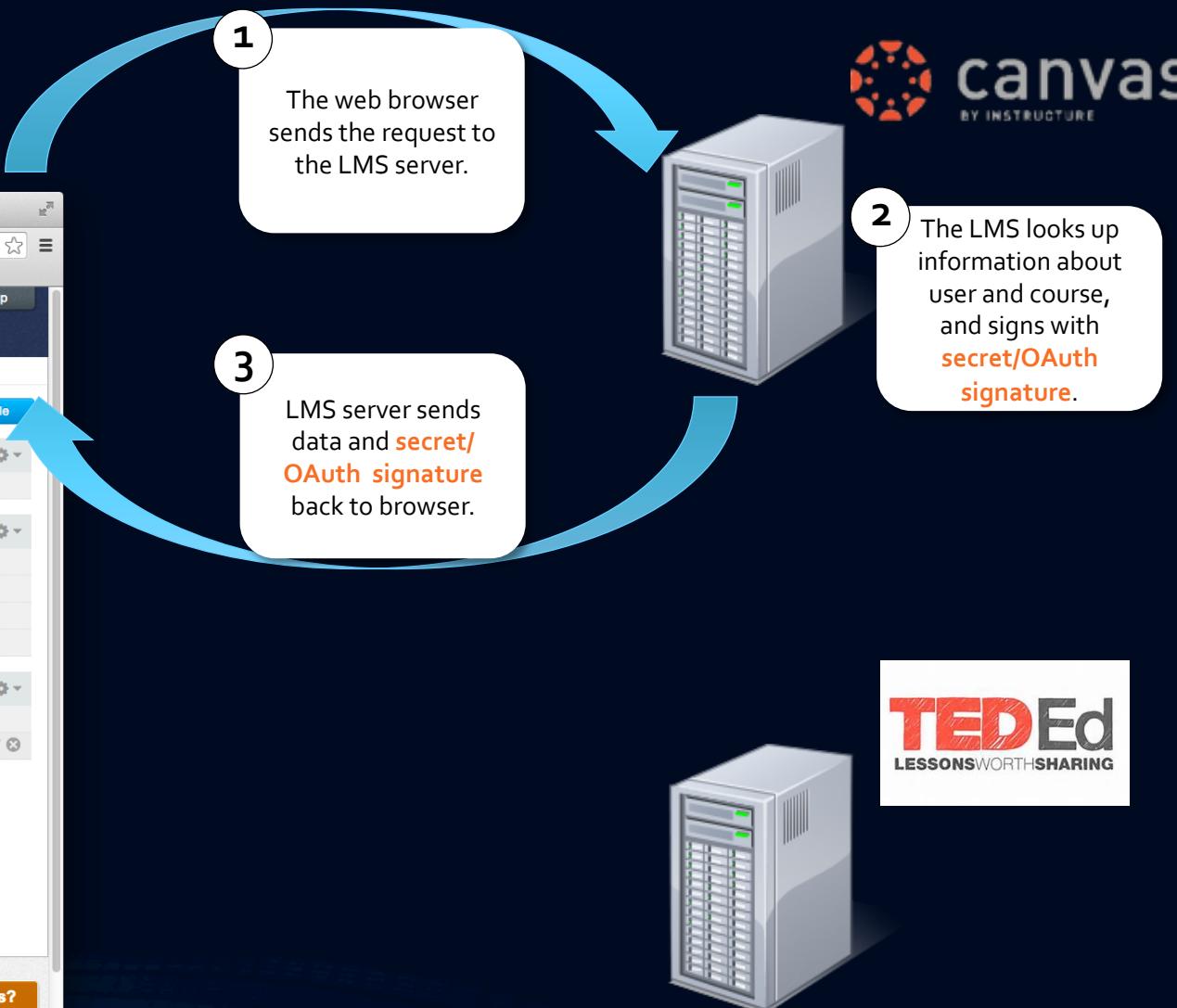
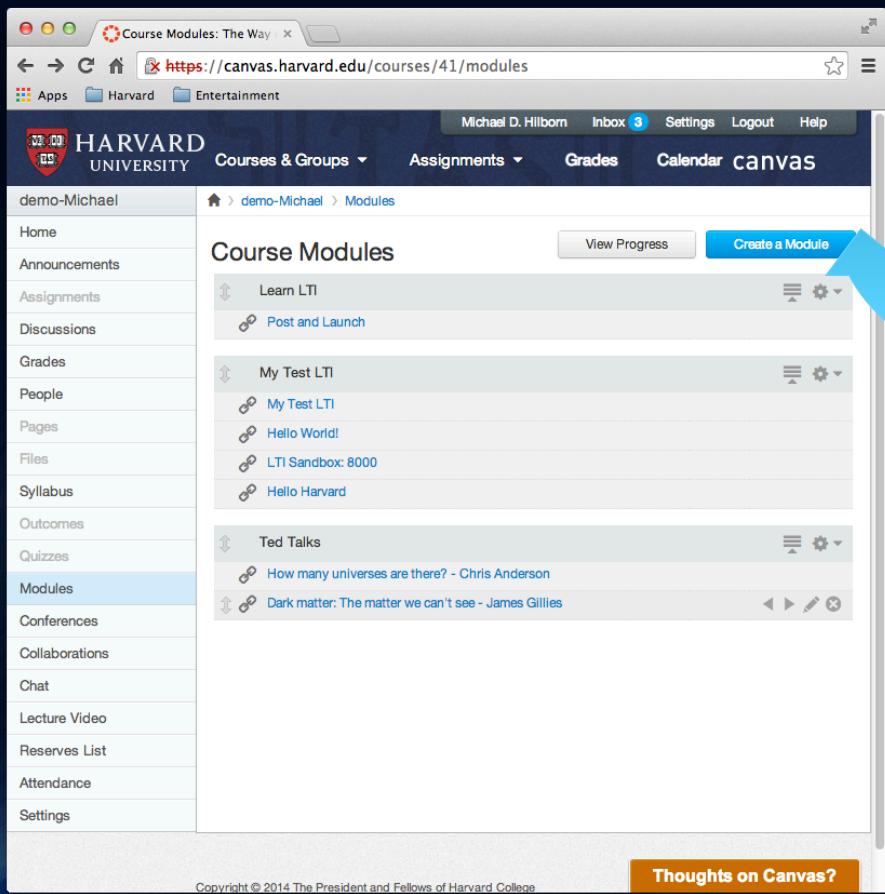
A screenshot of a web browser window titled "Course Modules: The Way". The URL is <https://canvas.harvard.edu/courses/41/modules>. The browser's address bar shows "https://canvas.harvard.edu/courses/41/modules". The page header includes the Harvard logo, user name "Michael D. Hilborn", and navigation links for "Inbox 3", "Settings", "Logout", and "Help". The main content area displays "Course Modules" with sections for "Learn LTI", "Post and Launch", "My Test LTI", "Ted Talks", and "Dark matter: The matter we can't see - James Gillies". In the "My Test LTI" section, there is a link labeled "Post and Launch". A yellow arrow originates from a blue starburst graphic containing the word "Click" and points towards this link. At the bottom of the page, there is a footer with the text "Copyright © 2014 The President and Fellows of Harvard College" and a "Thoughts on Canvas?" button.



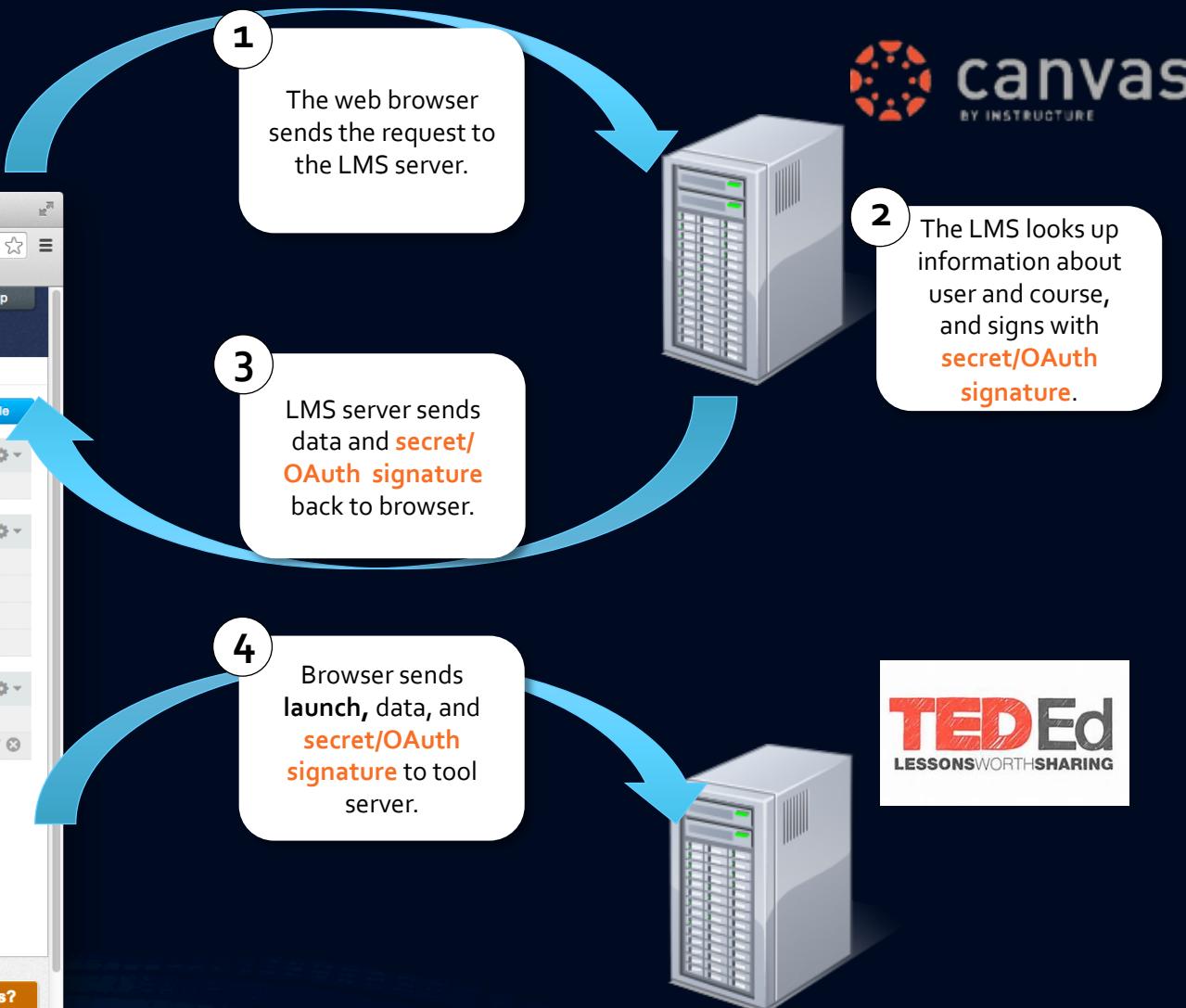
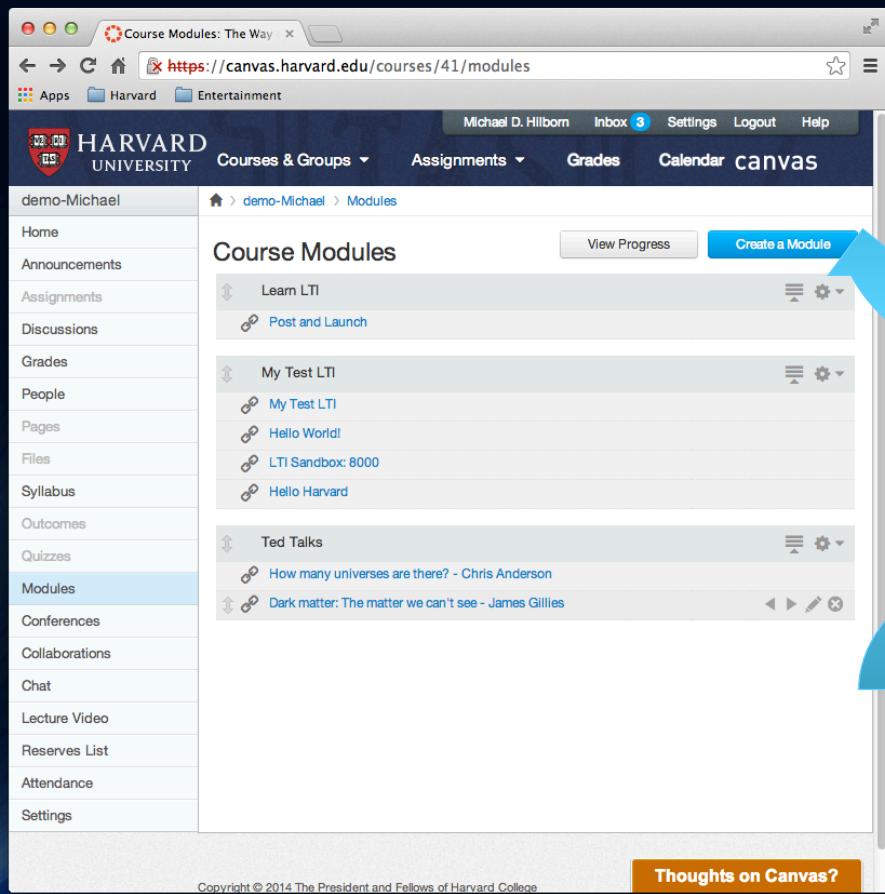
Basic LTI Launch – What The Web Experiences



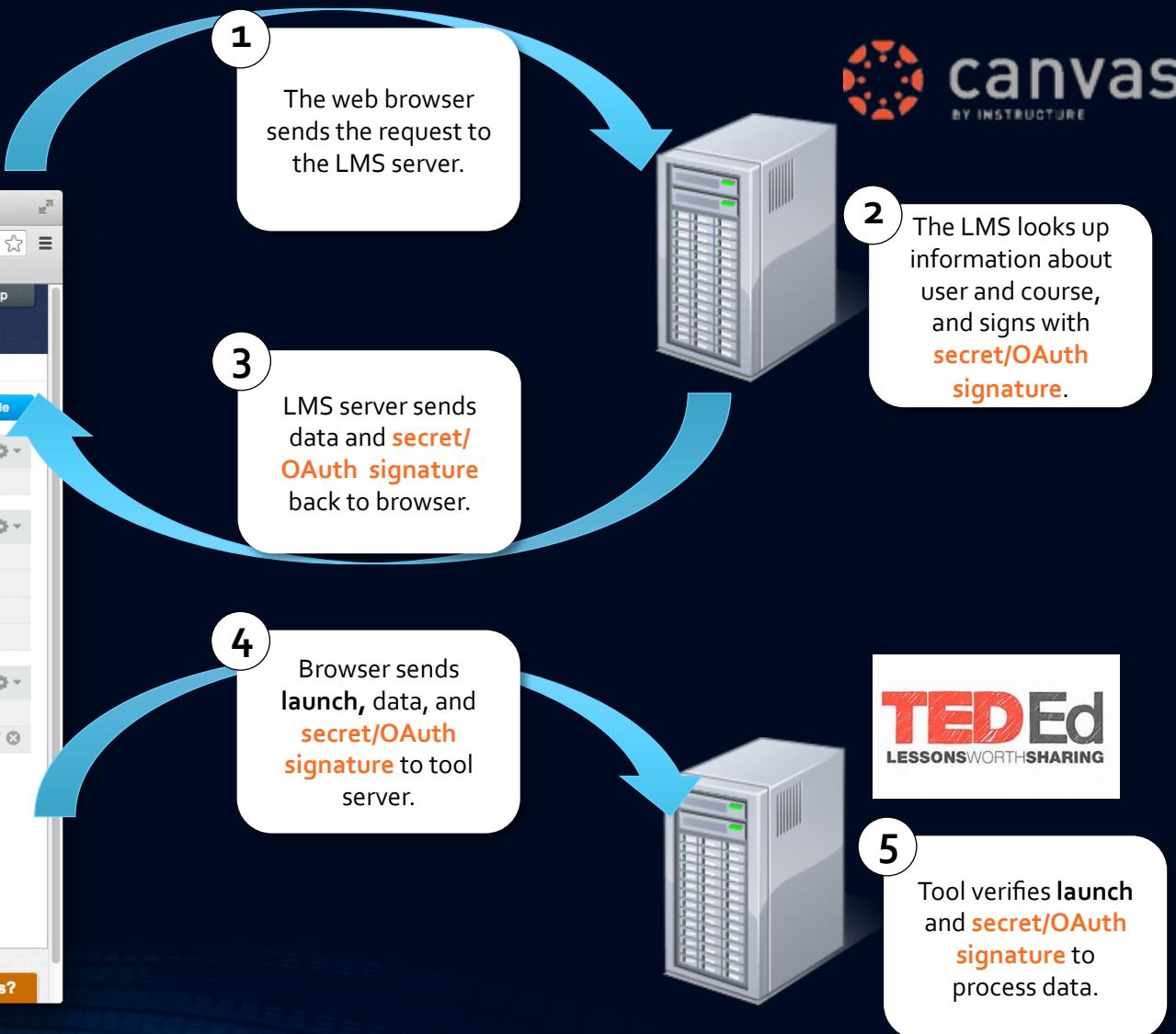
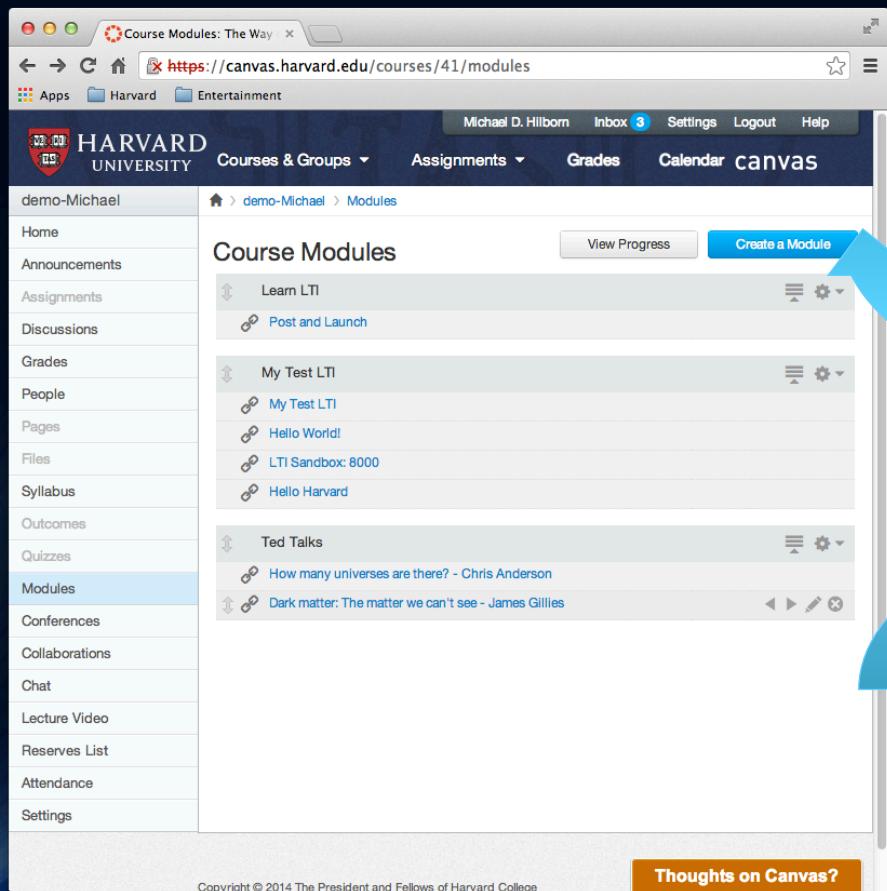
Basic LTI Launch – What The Web Experiences



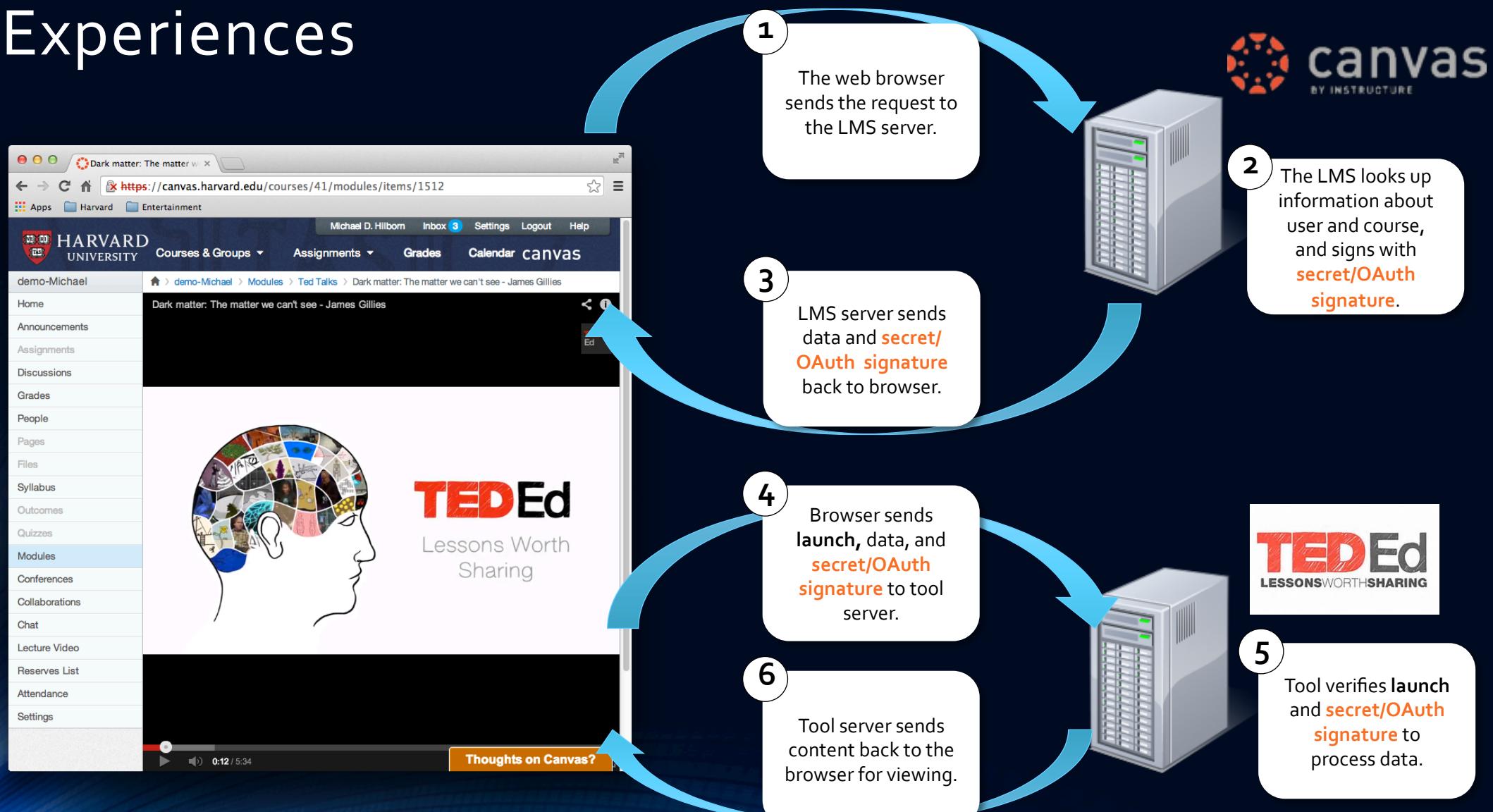
Basic LTI Launch – What The Web Experiences



Basic LTI Launch – What The Web Experiences



Basic LTI Launch – What The Web Experiences



Basic LTI Launch – Launch Parameters

- When an LMS launches a tool, it must send the following parameters:
 - **lti_message_type** set to **basic-lti-launch-request**
 - **lti_version** set to **LTI-1p0**
 - **resource_link_id** referencing a unique id of a tool instance
- Remember, these are just POST parameters
- If a tool does not receive these parameters, it should gracefully fail



Houston, we are good to go with a **basic-lti-launch-request**.

Basic LTI Launch – OAuth

- We have to ensure that the launch parameters and other data are coming from the LMS
- We do so by using OAuth signatures
- But what, exactly, is OAuth?

Didn't I tell you
not to accept
that message
without OAuth?

But OAuth
seemed so
complicated!



Basic LTI Launch – A Brief OAuth Primer

- OAuth is a **framework** that defines a method for clients (i.e. web apps) to access server resources (web service) on behalf of a resource owner (i.e. end-user)
- The end game of OAuth is an **access token** that the client may use to access **protected resources** on the server until/unless it is **revoked**
- What does that mean?
 - Generic methodology for API authentication.
 - Resource owner's credentials are protected (**not shared with client**)
 - Requests are protected by digital signatures.



Basic LTI Launch – A Brief OAuth Primer (cont'd)

- LTI uses a **subset** of OAuth for protecting HTTP requests
- A **digital signature** is a scheme designed to give the receiver of a request reason to believe that it is valid
- Signatures help **authenticate** LMS/Tool requests and prevent unauthorized requests:
 - Ex: protects the requests from tampering with the POST params
 - Ex: protects the LMS/Tool from replay attacks (repeated POSTs)
- For this reason, LTI requests should be **signed** and **verified**



Basic LTI Launch – A Brief OAuth Primer (cont'd)

ua = user/context attributes

secret = shared secret

sig = oauth_sig(ua, secret)



Basic LTI Launch – OAuth Parameters

- So, when an LMS launches a tool, it must send the following parameters:
 - **oauth_callback** often set to **about:blank**
 - **oauth_consumer_key** set to the consumer key
 - **oauth_nonce** set to the nonce generated by the LMS
 - **oauth_signature** set to the signature sent by the LMS
 - **oauth_signature_method** referencing the encryption (must use at least HMAC-SHA1)
 - **oauth_timestamp** representing the time/date of the request
 - **oauth_version** set to version of OAuth (such as **1.0**)
- Coupled with the **secret** that is shared by the LMS and tool, the tool must verify that the signature is valid before proceeding.

Basic LTI Launch – Recommended Parameters

- Although not required, it is recommended that these parameters be sent to an LTI tool:
 - **user_id** referencing unique id of current user
 - **roles** set to the users role, such as **Learner** or **Instructor**
 - **lis_person_name_full** set to the user's full name
 - **lis_person_name_given** set to the user's first name
 - **lis_person_name_family** set to the user's last name
 - **resource_link_title** referencing the name of the tool instance

Basic LTI Launch – Recommended Parameters (cont'd)

- There's more...
 - **context_id** referencing unique id of the course
 - **context_title** set to the name of the course
 - **context_label** set to the description of the course
 - **tool_consumer_info_product_family_code** set to the name of the LMS
 - **tool_consumer_info_version** set to the version number of the LMS
 - **tool_consumer_instance_guid** set to the instance of the LMS
 - **tool_consumer_instance_name** set to the name of the instance of the LMS
 - **tool_consumer_contact_version** set to the email of the LMS
- And if that weren't enough...

Basic LTI Launch – Custom Parameters

- Any number of custom values can be optionally sent to the tool
- The parameters should start with **custom_**
- Such as:
 - **custom_canvas_user_id**
 - **custom_canvas_user_login_id**

So many parameters! I can't keep up!

Hold onto your hat,
there's even more!
But you can find the
references online.



Basic LTI Launch – A Review

- To start an external LTI tool, the LMS must first **launch** the tool
- A launch consists of two items sent to the tool:
 - An OAuth signature
 - A set of parameters (a few required, many recommended, others custom)
- In other words, a **signed HTTP POST** request

And now that we
know what an LTI
tool should expect...



...You can see us
going about writing
a tool!



Writing An LTI Tool

IN WHICH WE DEMONSTRATE HOW TO WRITE A SIMPLE "HELLO WORLD" TOOL



First, Some Tools Of The Trade

- Choose a language
 - (We're choosing **PHP** for our demo)
- Install a Basic LTI library
 - (Here are a few: <http://www.edu-apps.org/code.html>)
- Set up a development environment
 - (We're using **Vagrant** and **VirtualBox**)
- Find an LMS that supports LTI
 - (We're using **Canvas**)



Examining The Code

- First, we include the Basic LTI library in our code

```
<?php
require_once 'ims-blti/blti.php';

// Any other setup code you might want

$context = new BLTI("secret", false, false);

header('Content-Type: text/html; charset=utf-8');
?>

<!DOCTYPE html>
<html>
    <!-- HTML goes here -->
<?php
    if ($context->valid) {
?>
        <h2>Hello LTI!</h2>
<?php
    } else {
?>
        <h2>This was not a valid LTI launch</h2>
<?php
    }
?>
    <!-- More HTML goes here -->

</html>
```

Examining The Code

- First, we include the Basic LTI library in our code
- Then we initialize the Basic LTI object...

```
<?php
require_once 'ims-blti/blti.php';

// Any other setup code you might want

$context = new BLTI("secret", false, false);

header('Content-Type: text/html; charset=utf-8');
?>

<!DOCTYPE html>
<html>
    <!-- HTML goes here -->
<?php
    if ($context->valid) {
?>
        <h2>Hello LTI!</h2>
<?php
    } else {
?>
        <h2>This was not a valid LTI launch</h2>
<?php
    }
?>
    <!-- More HTML goes here -->

</html>
```

Examining The Code

- First, we include the Basic LTI library in our code
- Then we initialize the Basic LTI object...
- ...with our shared OAuth secret

```
<?php
require_once 'ims-blti/blti.php';

// Any other setup code you might want

$context = new BLTI("secret", false, false);

header('Content-Type: text/html; charset=utf-8');
?>

<!DOCTYPE html>
<html>
    <!-- HTML goes here -->
<?php
    if ($context->valid) {
?>
        <h2>Hello LTI!</h2>
<?php
    } else {
?>
        <h2>This was not a valid LTI launch</h2>
<?php
    }
?>
    <!-- More HTML goes here -->

</html>
```

Examining The Code

- First, we include the Basic LTI library in our code
- Then we initialize the Basic LTI object...
- ...with our shared OAuth secret
- That's it, really... If the LMS sends this tool the proper LTI and OAuth parameters...

```
<?php
require_once 'ims-blti/blti.php';

// Any other setup code you might want

$context = new BLTI("secret", false, false);

header('Content-Type: text/html; charset=utf-8');
?>

<!DOCTYPE html>
<html>
    <!-- HTML goes here -->
<?php
    if ($context->valid) {
?>
        <h2>Hello LTI!</h2>
<?php
    } else {
?>
        <h2>This was not a valid LTI launch</h2>
<?php
    }
?>
    <!-- More HTML goes here -->

</html>
```

Examining The Code

- First, we include the Basic LTI library in our code
- Then we initialize the Basic LTI object...
- ...with our shared OAuth secret
- That's it, really... If the LMS sends this tool the proper LTI and OAuth parameters...
- ...you can check to see if the launch was valid; if so, write your code!

```
<?php
require_once 'ims-blti/blti.php';

// Any other setup code you might want

$context = new BLTI("secret", false, false);

header('Content-Type: text/html; charset=utf-8');
?>

<!DOCTYPE html>
<html>
    <!-- HTML goes here -->
<?php
    if ($context->valid) {
?>
        <h2>Hello LTI!</h2>
<?php
    } else {
?>
        <h2>This was not a valid LTI launch</h2>
<?php
    }
?>
    <!-- More HTML goes here -->

</html>
```

Examining The Code

- First, we include the Basic LTI library in our code
- Then we initialize the Basic LTI object...
- ...with our shared OAuth secret
- That's it, really... If the LMS sends this tool the proper LTI and OAuth parameters...
- ...you can check to see if the launch was valid; if so, write your code!
- Otherwise, fail gracefully

```
<?php
require_once 'ims-blti/blti.php';

// Any other setup code you might want

$context = new BLTI("secret", false, false);

header('Content-Type: text/html; charset=utf-8');

?>

<!DOCTYPE html>
<html>
    <!-- HTML goes here -->
<?php
    if ($context->valid) {
?>
        <h2>Hello LTI!</h2>
<?php
    } else {
?>
        <h2>This was not a valid LTI launch</h2>
<?php
    }
?>
    <!-- More HTML goes here -->

</html>
```

Is It Really That Simple?

- No, of course not
- Which brings us to our next topic...

Sometimes the
simplest solution is
the best answer...

Not in this case. The
truth is out there...



Challenges

IN WHICH WE DISCUSS THE CHALLENGES OF IMPLEMENTING LTI
TOOLS



Challenges – The Learning Curve

- It takes time to learn things
 - Basic LTI specifications
 - OAuth specifications
 - Your favorite LMS specifications
 - Your favorite Basic LTI library

I know LTI...

Show me.



Challenges – Setting Up (and Privacy)

- It takes time to set things up
 - Your development environment
 - Your production environment
 - (You really want to put this behind SSL)
- And speaking of security...
- ...Your tool still needs to comply with privacy
 - FERPA, HIPAA, etc.

Challenges – Two-Way Communication

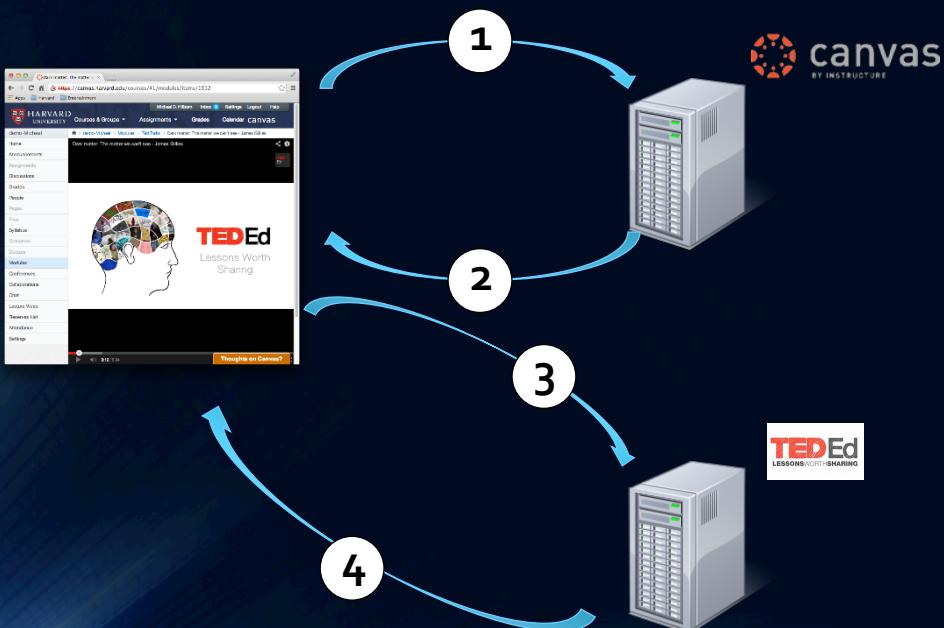
- So far, we've been talking about Basic LTI (v1.0)
- Basic LTI describes a launch...
- But it's mostly a one-way trip: The tool generally behaves independently in an iFrame
 - (There is a way to send messages back to the LMS after the tool has finished its job)



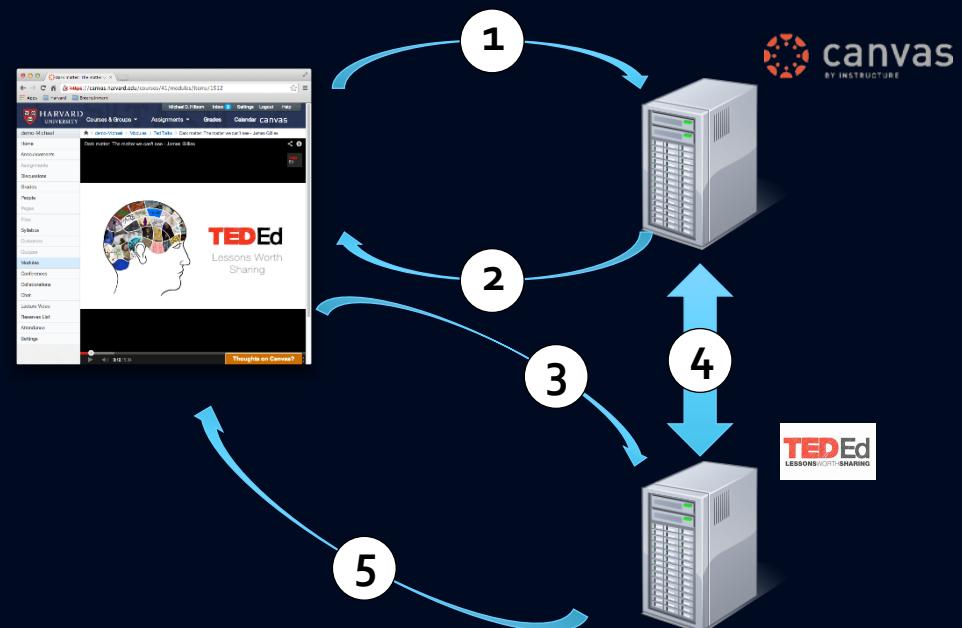
What do you
mean we're not
coming back?!

Challenges – Two-Way Communication

SO WE HAVE THIS...



WOULDN'T IT BE GREAT TO
HAVE THIS...?



Challenges – Two-Way Communication

- LTI v1.1 does support sending outcomes (grades) back to the LMS
- But that's about all
- You can, of course, have the tool use the LMS extensions and APIs...
- ...But that brings us back to rewriting code for each LMS!

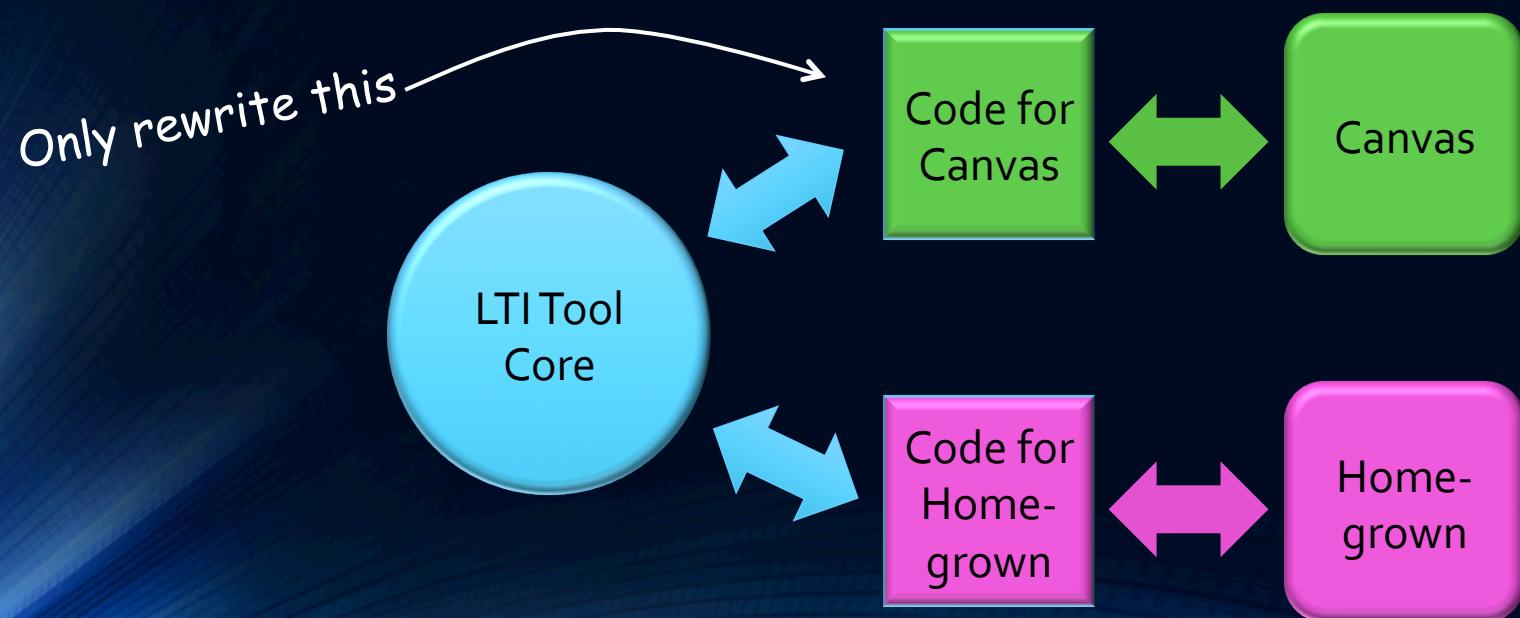
Geez, we're right back to where we started!

Darn it! This seems to happen every episode!



Abstract Your Code

- Fortunately, there are ways to minimize re-writing
- Abstract any portion of code that interacts with an LMS
- That way you need only have to rewrite one portion of your code if you switch to another LMS



LTIv2.0

- Utilizes REST-level 3 web services model for two-messaging between the LMS and your tool
- But it was just released in January 2014
- Doesn't seem to be extensively supported yet
 - (So we won't discuss it, but it's out there)

Perhaps LTI v2.0 will be the answer to our LMS woes.

At the very least, it will be progress...



Resources

IN WHICH WE LIST A VARIETY OF URLs THAT CAN HELP YOU OUT



Introductions To LTI

- The Official IMS LTI Page
<http://www.imsglobal.org/lti/>
- LTI Tutorial
<http://developers.imsglobal.org/tutorials.html>
- Briefing Paper on LTI
<http://publications.cetis.ac.uk/wp-content/uploads/2012/05/LTI-Briefing-Paper.pdf>
- Edu Apps – An open collection of learning tools built on LTI
<http://www.edu-apps.org/index.html>

Developing For LTI

- IMS Developers Web Site
<http://developers.imsglobal.org/>
- Writing LTI Stuff (with links to libraries)
<http://www.edu-apps.org/code.html>
- Canvas Dev and Friends – A Course for Canvas Development
<https://canvas.instructure.com/courses/785215>

OAuth Resources

- OAuth Web Site
<http://oauth.net/>
- The OAuth 1.0 Guide
<http://hueniverse.com/oauth/guide/>
- OAuth for Beginners
<http://blog.bittercoder.com/2008/06/10/oauth-for-beginners/>
- Introduction to OAuth
<http://blog.varonis.com/introduction-to-oauth/>

Vagrant and VirtualBox

- Oracle VirtualBox
<https://www.virtualbox.org/>
- Vagrant
<http://www.vagrantup.com/>
- A beginner's guide to Vagrant
<http://www.erikaheidi.com/2013/07/02/a-begginers-guide-to-vagrant-getting-your-portable-development-environment/>
- Automating Development Environments with Vagrant and Puppet
<http://blog.kloudless.com/2013/07/01/automating-development-environments-with-vagrant-and-puppet/>

Kudos

IN WHICH THANK THE MANY PEOPLE WHO MADE THIS
PRESENTATION POSSIBLE



Kudos

- Our colleagues in the Academic Technology Group
- Our colleagues in the iCommons Group
- IMS Global Learning Consortium
- And all of you!

Questions?

IN WHICH WE END OUR PRESENTATION AND ATTEMPT TO
ANSWER YOUR QUESTIONS



Answers!

IN WHICH WE REVEAL THE IDENTITY OF THE CHARACTERS IN
OUR PRESENTATION





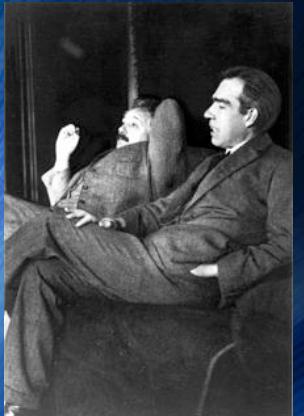
Nick Carraway & Jay Gatsby



Sherlock Holmes & Dr. Watson



Luke Skywalker & Yoda



Albert Einstein & Niels Bohr



Daedalus & Icarus



Thelma & Louise



Lucy & Ethel



C.S. Lewis &
J. R. R. Tolkien



Laurel & Hardy



Susan B. Anthony
&
Elizabeth Cady Stanton



Dana Scully & Fox Mulder



Neo & Morpheus



Gilligan & The Skipper