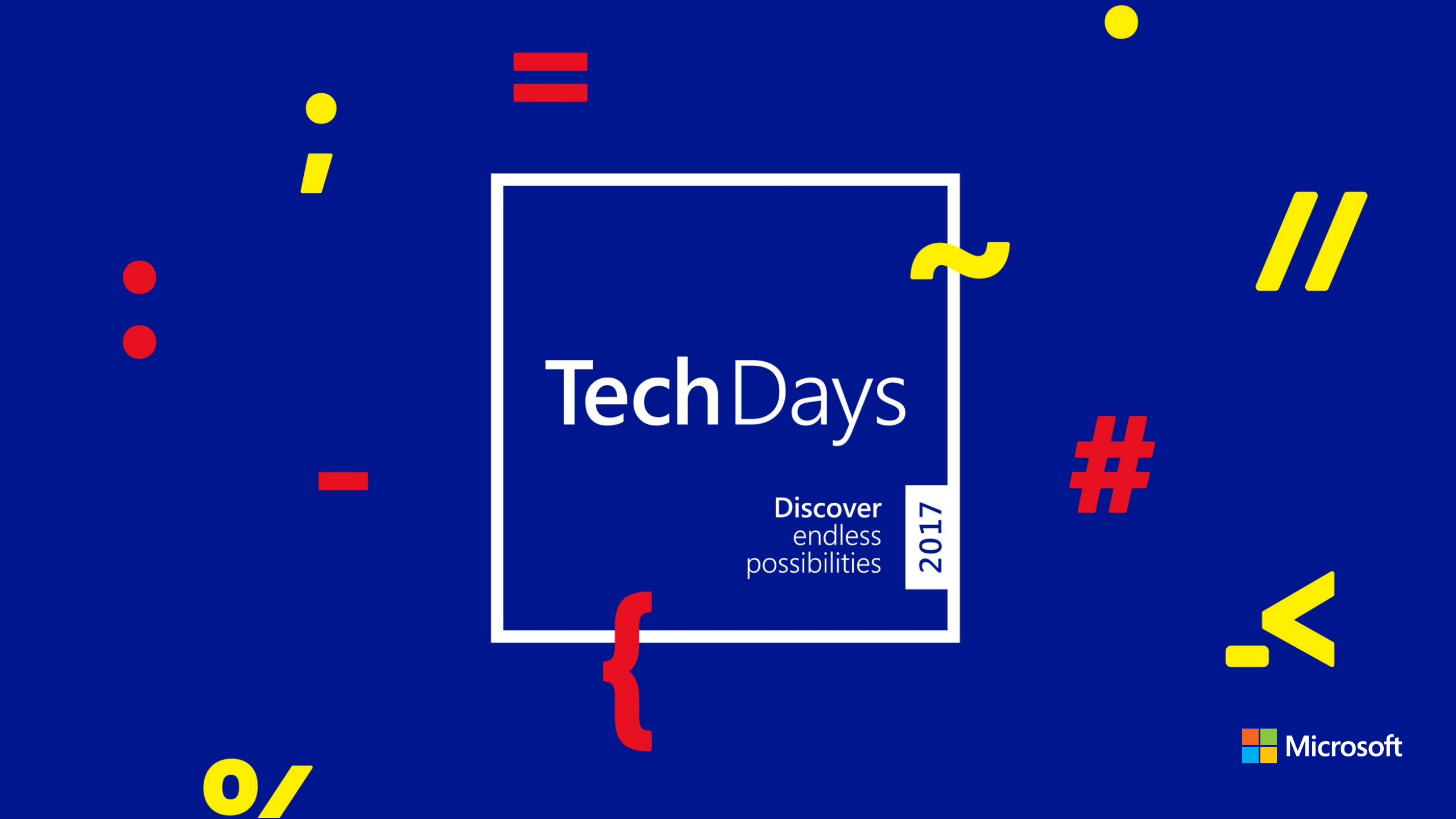

TechDays
Discover
endless
possibilities
2017
Microsoft

Twitter
#techdaysnl

Analyzing and improving performance of Universal Windows Apps

Johan Laanstra
Microsoft Corporation

Microsoft

Johan Laanstra

jlaans@microsoft.com

http://www.johanlaanstra.nl

@jplaanstra

http://github.com/jlaanstra

Overview

Performance 101

Windows Performance Toolkit

Demos

What to expect in this talk

Best practices

Get familiar with WPT

Performance 101

Performance 101

The fastest code is the code you don’t run

Define your targets and your goals and measure

Performance 101

Keep the UI thread responsive
Optimize Xaml
Use x:Bind
Minimize the work during startup
Follow language-specific recommendations

Performance 101: Keep the UI thread responsive

- CoreDispatcher.RunAsync
- CoreDispatcher.RunIdleAsync
- SynchronizationContext.Current
- Task.Run

Performance 101: Optimize Xaml

- Reduce element count
  - Reduce elements that don't produce pixels
  - Single panel is faster than nested panels
  - Use built-in border properties
  - Use Grid for overlapping UI
- Use x:Load and x:DeferLoadStrategy
- Use UI virtualization
- Minimize overdrawing
- Avoid LayoutUpdated
- Avoid re-templating controls to change color

Performance 101: Use x:Bind

- Less memory and CPU overhead
- Avoid DataContext
- Use event bindings/event handlers instead of ICommand
- Use functions instead of IValueConverter

Performance 101: Minimize the work during startup

- Defer work as long as possible
- Avoid unnecessary dll loading
- Call Windows.Current.Activate() as soon as possible
- Use SplashScreen.Dismissed for additional loading
- SplashScreen can be optional (Fall Creators Update)

```
<SplashScreen Optional="True">
```

- Opt-in to prelaunch

Performance 101: Follow language-specific recommendations

For C#:

- Don't call GC.Collect() manually
- Limit memory allocations
- Prefer managed types to avoid interop

For C++:

- Use latest C++ toolset
- Disable RTTI
- Avoid heavy use of ppl, consider co_await
- Prefer standard C++ over CX, consider C++/WinRT

https://docs.microsoft.com/en-us/windows/uwp/debug-test-perf/performance-and-xaml-ui

Windows Performance Toolkit

Windows Performance Toolkit

Windows Performance Recorder (WPR)

Allows you to capture a trace for the problem you want to investigate.

Windows Performance Analyzer (WPA)

Allows you to analyze the captured trace and do a detailed performance analysis of the problem to determine the root cause.

Approach

1. Identify the problem
2. Capture a trace of the scenario using WPR
3. Analyze the trace in WPA
   - Determine if the cause is CPU/disk/network
   - Identify the UI thread.
   - Analyze what time is spent where.
4. Modify the app and go back to 1.

Demo

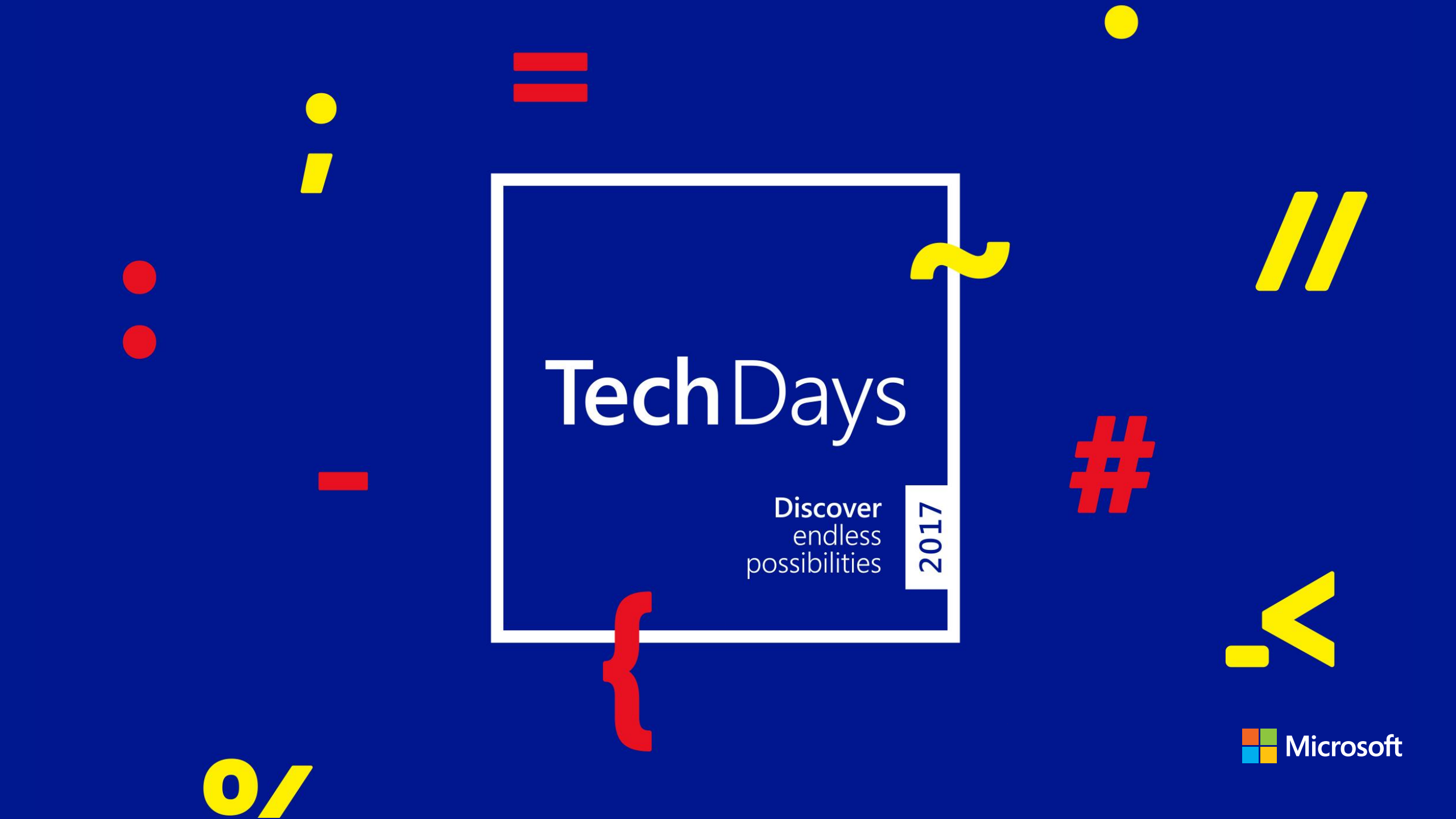
Adding custom tracing to your app

- Create an EventSource or LoggingChannel.
- Trace events at critical points in the app.
- Enable WPR to capture events from your new event source.

Demo

http://aka.ms/perftools

Microsoft


TechDays
Discover
endless
possibilities
2017
Microsoft