Delegated Ink Trails

Mario Bianucci
mabian@microsoft.com

https://aka.ms/DelegatedInkTrailExplainer
IRC: #delegated_ink_trail
Overview

• Reduce latency when inking

• Easy to use

• Minimal changes to app’s rendering pipeline
Why?

• Remote work and learning have both grown tremendously

Why?

• Good collaboration tools are key for both

• Low latency is critical for a good inking experience
Current enhancement issues

- Overlays have specific requirements to achieve

- Desynchronized canvas can’t synchronize removing and drawing

- `pointerevent.getPredictedEvents()` may not be accurate
Hardware & Operating System Input

Pointer moves

Web Browser

Browser process
OS Input arrives at the browser process.

Renderer process
App renders the stroke and calls the API.

GPU Process
Draws stroke onto the screen and draws a tail connecting the end of the stroke through the new points.

Display hardware

Cognition
We see the change.

new links & components
existing links
Further improvement

• OS Compositor introduces latency

• Some OS support for last minute drawing on app’s behalf

• Last minute drawing allows more real points to arrive
Skia Polyfill in Chromium

• Working today in Chromium
  • Drawn via Skia
  • Proof of concept, correctness

• Internal customers providing positive feedback
  • PowerPoint
  • Teams
  • Whiteboard
Delegated Ink Trails via OS

- Upcoming Windows 10 API support
- Internal testing is promising
Proposed WebIDL

partial interface Navigator {
    [SameObject] readonly attribute Ink ink;
};

interface Ink {
    Promise<DelegatedInkedTrailPresenter> requestPresenter(DOMString type,
    optional Element? presentationArea = null);
}

dictionary InkTrailStyle {
    DOMString color;
    unrestricted double diameter;
}

interface DelegatedInkedTrailPresenter {
    void updateInkedTrailStartPoint(PointerEvent evt, InkTrailStyle style);
    readonly attribute Element? presentationArea;
}
The Road Ahead

• Consensus on API
  • Internal partners have found it to be useful.
  • Is the current shape ideal for common use cases?
  • Potential issues?

• Taking it out of incubation
  • Is there support for standardizing?