Requirements for IndieUI: Events 1.0 and IndieUI: User Context 1.0

W3C First Public Working Draft 22 April 2014

This version:
http://www.w3.org/TR/2014/WD-indie-ui-requirements-20140422/
Latest published version:
http://www.w3.org/TR/indie-ui-requirements/
Latest editor's draft:
https://dvcs.w3.org/hg/IndieUI/raw-file/default/src/indie-ui-requirements.html
Editor:
Michael Cooper, W3C

Abstract

This document outlines the requirements that the IndieUI Working Group has set for development of IndieUI: Events 1.0 and IndieUI: User Context 1.0. These requirements will be used to determine whether the IndieUI WG has met its goals as these specifications advance through the W3C Recommendation Track Process. This document introduces a series of user scenarios common to the two specifications, and a list of technical requirements needed to meet those scenarios for each specification. It also provides information about how the requirements are addressed. For background information on IndieUI: Events and IndieUI: User Context see the IndieUI Overview.

Status of This Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at http://www.w3.org/TR/.

This is a First Public Working Draft of "Requirements for IndieUI: Events 1.0 and IndieUI: User Context 1.0" from the Independent User Interface (Indie UI) Working Group. The Working Group intends to develop this document and publish it as a Working Group Note to support the

Recommendation-track deliverables IndieUI: Events 1.0 and IndieUI: User Context 1.0. "Requirements for IndieUI: Events 1.0 and IndieUI: User Context 1.0" addresses both those specifications, although in this version only requirements for IndieUI: Events have been elaborated. The document provides scenarios for web content interaction that need additional standards support to optimize interaction by people with disabilities, followed by requirements for the specifications to meet those scenarios. Where requirements are currently met by the specification, an appropriate link is provided.

Feedback on this document is essential to success of these technologies. The IndieUI WG asks in particular:

- Do the scenarios comprehensively describe situations that should be addressed in IndieUI: Events and IndieUI: User Context?
- Do the requirements sufficiently address the scenarios?
- Are there scenarios or requirements that should not be addressed in the 1.0 version of these specifications?

To comment on this document, send email to public-indie-ui-comments@w3.org (comment archive). Comments are requested by 23 May 2014. In-progress updates to the document may be viewed in the publicly visible editors' draft.

Publication as a First Public Working Draft does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.

This document was produced by a group operating under the 5 February 2004 W3C Patent Policy. The group does not expect this document to become a W3C Recommendation. W3C maintains a public list of any patent disclosures made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains Essential Claim(s) must disclose the information in accordance with section 6 of the W3C Patent Policy.

Table of Contents

1. Introduction

Scripting usable interfaces can be difficult, especially when one considers that user interface design patterns differ across software platforms, hardware, and locales, and that those interactions can be further customized based on personal preference. Individuals are accustomed to the way the interface works on their own system, and their preferred interface frequently differs from that of the web application author's preferred interface. Some complex web applications can provide a much better experience if given access to information such as a user's preferred color, font, screen, and even restricted assistive technology settings such as a preference to render captions, or whether a screen reader is on.

Custom interfaces often don't take into account users who access web content via assistive technologies that use alternate forms of input such as screen readers, switch interfaces, or speech-based command and control interfaces. For example, a web page author may script a custom interface to look like a slider (e.g. one styled to look like an HTML "range" input) and behave like a slider when using standard mouse-based input, but there is no standard way for the value of the slider to be controlled programmatically, so the control may not be usable without a mouse or other pointer-based input.

IndieUI: Events defines a way for web authors to register for these request events. Authors declaritively define which actions or behaviors a view responds to, and when it is appropriate for browsers to initiate these events. IndieUI: User Context provides authorized web applications access to information about a user's relevant settings and preferences, to provide the best possible user experience to all users. General web pages developed using best practices may never need access to restricted user settings, but complex web applications can utilize this information to enhance the performance and user interface.

One of the core principles behind these specifications is that that it operates on a backwards-compatible, opt-in basis. In other words, the web application author has to first be aware of these events, then explicitly declare each event receiver and register an event listener, or user agents behave as normal and do not initiate these events. If a web application does not respond to the event, the user agent may attempt fallback behavior or communicate to the user that the input has not been recognized.

For further background information on IndieUI: Events and IndieUI: User Context see the IndieUI Overview.

2. Use Cases and Scenarios

2.1 Manipulate a map

A person is using a map to find the location and layout of a local park

in a web-based mapping application so they can print it out, using their touch-screen laptop. They know the general location, and see the green area on the lower-left-hand corner of the map on the screen. They touch that part of the screen, and use a zooming gesture to center and zoom in on that section of the screen, then fine-tune the centering using the arrow keys on their keyboard and zoom in further using the context menu on their laptop's trackpad. Finally, they use a rotation gesture on the touchscreen to re-orient the map around the point of interest. Once they have the view they want, they use the browser's control menu to print the map.

2.2 Execute a popup

A user whose point of regard (focus) is on a UI object that support popups performs an action that causes the web application to render the popup. A popup could be a popup dialog box or a popup menu. The user would like to be made aware that either of these popup options are available and be able to cause the popup to render using a variety of device input interaction methods such as a keyboard command, a gesture, a voice command, or a right mouse click.

> NOTE
>
> WAI-ARIA provides a property, aria-haspopup, that indicates the UI Widget supports a popup.

2.3 Open or collapse a tree branch, menu, expandable grid cell, or expandable section

A user whose point of regard (focus) is on a UI object that indicates the UI object is expandable / collapsible to reveal / hide subordinate content would like to perform an action to cause the web application to reveal / hide the content. Common UI objects that support this function are tree items in tree controls, gridcells in treegrids that expand to reveal new rows, accordion tabs which reveal / hide panels of content, or expandable and collapsible regions (e.g. portlets). The user would like to be made aware that these options are available and be able to reveal / hide the content using a variety of device input interaction methods such as a keyboard command, a gesture, a voice command, or possibly a mouse click.

> NOTE
>
> WAI-ARIA provides an aria-expanded property that indicates that is expandable when it is set to false.

> NOTE

> WAI-ARIA has a container role of dialog that this could be applied to. Essentially this would be equivalent to an escape key .

2.4 Activate a UI component

A user whose point of regard (focus) is on a UI object, that can be activated, and would like to perform an action to activate it. Example UI objects that support activation are push buttons, radio buttons, checkboxes, and menu items. The action could be in the form of a tap, a gesture, a voice command, a mouse click, a keyboard key, or a command from an alternative input device.

> NOTE
>
> This has been proposed in the past and the legacy "click" was used instead - DOMActivate.

2.5 Move the focus point within a UI component down to the next / visual right / visual down, or previous / visual left / visual up item within a UI component

A user whose focus is on a UI object that supports next and previous navigation within the UI component would like to control the UI to move its current active item (usually rendered visibly as its point of regard) to the next or previous item within its internal navigation sequence. This might be the next or previous item within a listbox, tree widget, menu, menubar, or grid, treegrid, select, or any other type of UI Component supporting this function. Visually the next item is usually right or down, and the previous item is usually left or above. The action could be in the form of a gesture, a voice command, a right mouse click, a keyboard key, or an alternative input device. Some UI components may choose to force an item selection in response to the action.

> NOTE
>
> - We cannot assume we know all UI components capable of supporting this function.
> - Some UI components may move its internal point of regard back to the beginning of its navigation sequence after it reaches the end.
> - This movement is not response to a tab notification which should be designed to move among widgets and interactive controls that are not managed by another UI component.
> - This would not be applicable to Dialogs or form elements that intended to be in the tab sequence.
> - "Visual right" vs "next element" / "visual left" vs "previous element" are two different things. Right and left are locale-independent. Next / previous element would change direction based on locale, for example, in RTL languages like Arabic.

- We cannot assume that moving the point of the regard to the right will not move the point of regard back to the left-most managed object when starting from the right-most managed UI object.

2.6 Direct a media player to start, stop, or pause playing

A user whose point of regard (focus) is within a media player would like to notify the application to start, stop, or pause the playing of the video, audio, or animation. When a notification to start, stop, or pause the player is received the rendering starts, stops, or pauses. Start starts playing form the current time in the media play sequence. Stop moves the current time in the media play back to the start of the media sequence. Pause stops playback but does not move the time pointer. Example UI components would be a video player or audio player. The action could be in the form of a gesture, a voice command, a keyboard key combination, or an alternative input device.

> NOTE
>
> Do we want to include animation such as SVG animation? This would require the user agent to respond to the notification.

2.7 Toggle rendering captions

A user whose point of regard (focus) is within a media player would like to notify the application to toggle the rendering of video or audio captions. When a notification to toggle captions is received the player the toggles the rendering of the caption track at the synchronization point within the video or audio stream play. Example UI components would be a video player or audio player. The action could be in the form of a gesture, a voice command, a keyboard key combination, or an alternative input device.

> NOTE
>
> HTML5 has a a new media controller API that could be used to facilitate this happening.

2.8 Increase or decrease the volume

A user whose point of regard (focus) is within a media player would like to notify the application to increase or decrease the video or audio volume. When a notification to increase or decrease the volume is received the player increases or decreases the rendering volume. Example UI components would be a video player or audio player. The action could be in the form of a gesture, a voice command, a keyboard key combination, or an alternative input device.

NOTE

- The increments would need to depend on the granularity of the player
- Do we want to be able to change the volume increments?
- Should we make this more generic to increase a control - such as a modifiable range object (slider)?

2.9 Zoom in or out

A user whose point of regard (focus) is within a zoomable object would like to notify the application to zoom in on the object. When a notification to zoom in a particular factor is received, the object zooms in by that factor and optionally provides more detail. Example UI components would be an SVG rendering of a CAD drawing or a key component of a scatter plot. The notification could be in the form of a gesture, a voice command, a keyboard key combination, or an alternative input device.

NOTE

- Do we want to separate Drill Down from Zoom Out or should we leave that up to the applications? A drill down might provide more information. An end user may think zoom simply enlarges an area vs. providing more detail.
- Is this distinct in any way from the zoom operation discussed in the map scenario? Should we just have zoom, pan and rotate as separate scenarios (or even just a single combined scenario)?

2.10 Pan right, left, up, or down

A user whose point of regard (focus) is within a media player would like to direct the ui object to pan up, down, left, or right so that more information can be revealed to the user in the direction of the pan. Example UI components would be a chart, subway map, CAD drawing, etc. The action could be in the form of a gesture, a voice command, a keyboard key combination, or an alternative input device.

NOTE

Do we want to include animation such as SVG animation? This would require the user agent to respond to the notification.

2.11 Move point of regard to the beginning or end of its navigation sequence

A user whose point of regard is rendered within a focused UI Object that manages its own navigation and would like to direct the ui object to move the point of regard to the beginning or end of the navigation sequence similar to a "Home" or "End" button. Example UI components would be a listbox, video player, audio player, tree widget, contenteditable area, tree widget, gridd, treegrid, or tablist. The action could be in the form of a gesture, a voice command, a keyboard key combination, or an alternative input device.

2.12 Grab, move, and release grabbable UI object

A user whose point of regard is non a UI object would like to grab the object for the purposes of moving it such as in a drag operation. After moving the point of regard, they drop the currently grabbed object on the object with focus at the current point of regard location. Example UI components to grab and move would be a light box, a lisbox item, a tree item, or a drawing object. Example UI components on which an item could be dropped would be a light box empty box or a line indicating a location between light box items, a lisbox item to drop the item before or after the current listbox item, a tree item to add a new item in a subtree, or a region of the web application or a drawing object. The action could be in the form of a gesture, a voice command, a keyboard key combination, or an alternative input device.

> NOTE
>
> - WAI-ARIA has an aria-grabbed property such that when set to false indicates that it may be grabbed for the purposes of moving it.
> - WAI-ARiA defines roles for objects such as regions
> - Should this be left up to the user agent?
> - Should the drop occur if the target does not have aria-dropeffect set?

2.13 Continuous or discontinuous multi-slection within a UI object

A user whose point of regard is on a UI object would like to select multiple continuous or discontinuous items within a supporting UI control. Once initiated it would tell the UI component to start a run of either continuous or discontinuous item selections with in the UI control. For continuous selection, as the user navigates the items within the UI Object container each item navigated to is automatically selected until the multi-selection process terminates. For discontinuous selection, a separate command would be given to actually select individual items within the UI object but navigation among items in the UI object would not cause an actual selection to occur. When selection is complete, the user directs the UI object either to save the currently selected items and exit the selection process, or to cancel the selection process and clear the selection. Examples are options in a listbox, gridcells withing a gridd, or treeitems within a tree. This action could

be in the form of a gesture, a voice command, a keyboard key combination, or an alternative input device.

> NOTE
>
> WAI-ARIA has an aria-selected state that can be used to reflect the selected state of the item within a UI object.

2.14 Increase or Decrease Size of UI object by a small or large increment

A user whose point of regard is on a UI object would like to ask it to increase or decrease its size by a small or large increment. This is very common in drawing objects such as drawing objects in a flow diagram or presentation tool where the user is attempting to create a visual UI. For people with mobility impairments this is very hard to do with a pointing device and alternate forms of input are necessary. This action could be in the form of a gesture, a voice command, a keyboard key combination, or an alternative input device.

> NOTE
>
> We should provide an ARIA property for ARIA 2.0 that conveys that an object is resizeable.

2.15 Move a UI object

A user whose point of regard is on a UI object would like to be able to move the object in different directions by both small and large ingrements. This is very common in drawing objects such as drawing objects in a flow diagram or presentation tool where the user is attempting to create a visual UI. For people with mobility impairments this is very hard to do with a pointing device and alternate forms of input are necessary. This action could be in the form of a gesture, a voice command, a keyboard key combination, or an alternative input device.

> NOTE
>
> - We should provide an ARIA property for ARIA 2.0 that conveys that an object is moveable.
> - We need to decide how we can do this in terms of the number of different types of events we use. This requires more group discussion.

2.16 Reveal or hide additional text without selection

User indicates that further information about an object is desired and should be shown, or once seen, no longer needed and should be hidden. This further information may be a hint similar to Hover on a mouse based system. Content may be timed or something that completes display. This action could be in the form of a gesture, a voice command, a keyboard key combination, or an alternative input device.

- Should we use a hover device independent event instead and leave this up the developer?
- This is related to S32, S33, and S34

2.17 Change nearness indication without selection

User safely tests whether an object is active and can be selected. This further information may be a change in visual appearance, but not an actual selection or activation. (Similar to MouseOver on a mouse based system)

> NOTE
>
> - Should we use a hover device independent event instead and leave this up the developer?
> - This is related to S31, S32, and S34

2.18 Cause a non-selection active state indicator to disable

The result is an a dormant state indication. (Similar to Mouse Exit on a mouse based system)

> NOTE
>
> - Should we use a hover device independent event instead and leave this up the developer?
> - This is related to S31, S32, and S34

2.19 Cause a suspend/resume of live updates to the page. Can very from live regions, live blogging, twitter stream, live download.

> NOTE
>
> Discussed at November 1, 2012 TPAC Face to Face?

3. Events Requirements

Requirements in this section are expected to be met by IndieUI: Events.

3.1 API

Provide an API layer to define user intended events that is agnostic of the specific input methodology and independent of a user's particular platform, hardware, locale, and preferences.

Addresses scenario(s):

Met by: Goals

3.2 Input Device Independence

Allow the API to support user commands without requiring specific physical events.

Addresses scenario(s):

Met by: Goals

3.3 Interaction Independence

Do not require specific physical user interactions (keyboard combinations, gestures, speech, etc.) to trigger particular IndieUI events.

Addresses scenario(s):

Met by: Scope

3.4 Backwards Compatibility

Structure the events such that they are only triggered if the application registers an interest in them, to optimize performance and allow backwards compatibility.

Addresses scenario(s):

Met by: Backwards Compatibility

3.5 Events Handled

Provide a way for applications to communicate that a given event request was or was not handled so the host OS or UA can attempt fallback behaviour.

Addresses scenario(s):

Met by: Backwards Compatibility

3.6 Non-Interference

Do not block standard events when listening for IndieUI events.

ISSUE-15 may impact this. Editorial Note

Addresses scenario(s):

Met by: Backwards Compatibility

3.7 Reset

Provide a way to "reset" ui-actions on descendant node.

ISSUE-15 may impact this. Editorial Note

Addresses scenario(s):

Met by: Backwards Compatibility

3.8 Event Delegation

Allow event delegation without affecting performance and scoping of events.

ISSUE-16 may impact this. Editorial Note

There may be additional requirements related to Section 2 UI Actions after we clarify implications of this structure. Editorial Note

Addresses scenario(s):

Met by: UI Actions

3.9 Event Order

There will be a requirement for how IndieUI events fit in with the order of other events.

This needs rewording. Editorial Note

ISSUE-15 may impact this. Editorial Note

Addresses scenario(s):

Met by: UI Request Events

3.10 Physical Event Association

Might need a requirement to be able to associate an IndieUI event with other related physical events

This needs rewording. Editorial Note

ISSUE-15 may impact this. Editorial Note

Addresses scenario(s):

Met by: UI Request Events

3.11 Extend UI Events

IndieUI Events must extend UIEvents unless the requirements are met directly in UI Events.

Addresses scenario(s):

Met by: UIRequestEvent

3.12 User Input Functions

IndieUI must support the following functions unless supported by other technologies:

- undo
- redo
- expand
- collapse
- dismiss
- delete
- move
- pan
- rotation
- scroll
- valuechange
- zoom

Addresses scenario(s): 2.3 Open or collapse a tree branch, menu, expandable grid cell, or expandable section, 2.9 Zoom in or out, 2.10 Pan right, left, up, or down, 2.15 Move a UI object

Met by: UIRequestEvent

3.13 Keyboard and Mouse compatibility

The properties of IndieUI request events must be a superset of the events from at least both keyboard and mouse events in the UI Events specification.

Addresses scenario(s):

Met by: UIRequestEvent

3.14 Linear Navigation

Support linear navigation for first, previous, next, and last.

- first
- previous
- next

- last

Addresses scenario(s): 2.5 Move the focus point within a UI component down to the next / visual right / visual down, or previous / visual left / visual up item within a UI component , 2.11 Move point of regard to the beginning or end of its navigation sequence?

Met by: UIFocusRequestEvent

3.15 Directional Navigation

Support directional navigation for:

- up
- down
- left
- right
- up and right
- up and left
- down and right
- down and left

Addresses scenario(s): 2.5 Move the focus point within a UI component down to the next / visual right / visual down, or previous / visual left / visual up item within a UI component

Met by: UIFocusRequestEvent

3.16 Non-Linear Navigation

Provide a mechanism for users to move focus non-linearly to other sections of the document such as toolbar, palette, and windows.

Action-57: also help (main covered by landmark) Editorial Note

Addresses scenario(s):

Met by: UIFocusRequestEvent

3.17 Landmark Navigation

Provide a way to navigate amongst landmark regions.

This may be at risk in 1.0 and could be pushed to future version. Might be a11y specific. Editorial Note

Addresses scenario(s):

Met by: UIFocusRequestEvent

3.18 Return to Previous Point of Regard

Provide a way for users to return to their previous point of regard, like emacs and vid and IDEs support.

Need to remove the tool-specific examples. Editorial Note

This does not have consensus yet. Need to determine if it is just keyboard or others as well. Could be pushed to future version. Seems to have general use cases. Editorial Note

Addresses scenario(s):

Met by: UIFocusRequestEvent

3.19 Manipulate Objects

Provide a mechanism to perform the following manipulations on objects or the screen:

- move
- pan
- rotate
- zoom

Addresses scenario(s): 2.9 Zoom in or out, 2.10 Pan right, left, up, or down, 2.15 Move a UI object

Met by: UIManipulationRequestEvent

3.20 Scroll

Provide a mechanism for custom scroll views to scroll the view in the following manners:

- by increments
- entire screens
- to the limit

in the following directions:

- up
- down
- left
- right

Addresses scenario(s):

Met by: UIScrollRequestEvent

3.21 Range Controls

Provide a mechanism to adjust numeric values of custom range controls by small and large increments, or to minimum and maximum values.

Addresses scenario(s):

Met by:

3.22 Feature Detection

Provide a mechanism for content authors to determine if the user agent has support for specific events.

Addresses scenario(s):

Met by: UIValueChangeRequestEvent

3.23 Context Menus

Provide a mechanism to access context menus and perhaps other secondary actions.

Addresses scenario(s): 2.2 Execute a popup

See ISSUE-3. This might be a future requirement. Editorial Note

3.24 Media Controls

Provide a mechanism for standard media controls including:

- play
- pause
- stop
- fast forward
- rewind
- move to start
- move to end
- increase volume
- decrease volume
- set volume to minimum
- set volume to maximum
- mute
- unmute
- move to next track
- move to previous track

See ACTION-16, ACTION-19, and ACTION-20. Some of these might be future requirements. Editorial Note

Addresses scenario(s): 2.6 Direct a media player to start, stop, or pause playing, 2.8 Increase or decrease the volume

Met by:

3.25 Suspend and Resume

Provide a mechanism to suspend or resume an activity

See ACTION-17. This might be a future requirement. Editorial Note

Addresses scenario(s): 2.19 Cause a suspend/resume of live updates to the page. Can very from live regions, live blogging, twitter stream, live

download.

Met by:

3.26 Selection

Provide a mechanism to select one or more objects (contiguously or discontiguously) for the purpose of performing an action.

See ACTION-25 Editorial Note

Contiguous and discontiguous selection might be separate requirements. Editorial Note

Addresses scenario(s): 2.13 Continuous or discontinuous multi-slection within a UI object

Met by:

3.27 No Implied Click

Provide a mechanism to activate an object without implying a click event.

See ACTION-53. Scenario is slide to unlock screen on FirefoxOS. Need to coordinate with Webapps. Editorial Note

Addresses scenario(s):

Met by:

3.28 Point of Regard

Provide an event that reacts to gain or loss of point of regard.

See ACTION-58. Still need a better name than Point of Regard. Editorial Note

Addresses scenario(s):

Met by:

3.29 Text Editing

Provide a mechanism to support text editing.

See ISSUE-9. This is probably a future requirement. Editorial Note

need to spell out specific events required To-do

Addresses scenario(s):

Met by:

3.30 Quick Search

Provide a mechanism to support quick search functionality directly within the web application.

See ISSUE-12. This is probably a future requirement. Editorial Note

Addresses scenario(s):

Met by:

3.31 Resize Object

Provide a mechanism to resize objects in graphical editing applications with ability to constrain proportions.

See ACTION-26. This is probably a future requirement. Editorial Note

Addresses scenario(s): 2.14 Increase or Decrease Size of UI object by a small or large increment

Met by:

3.32 Set Rotation Centerpoint

Provide a mechanism to set the centerpoint of a rotation request.

See ACTION-26. This is probably a future requirement. Editorial Note

Addresses scenario(s):

Met by:

3.33 Table Sort

Provide a mechanism to request grid sort by columns

See ACTION-31. This is probably a future requirement. Editorial Note

Addresses scenario(s):

Met by:

4. User Context Requirements

Requirements in this section are expected to be met by IndieUI: User Context.

This version of the document does not yet elaborate requirements for IndieUI: User Context. Editorial Note

5. Other Requirements

Requirements in this section address goals fo the IndieUI Working Group but are expected to be met by current and future work from other efforts. They are included here to help provide a complete picture of the requirements space addressed by the IndieUI Working Group.

A. Acknowledgements

A.1 Active IndieUI Members

At the time of publishing, the following individuals were active, participating members of the IndieUI Working Group.

- Michael Cooper, Staff Contact, W3C Staff
- James Craig, Apple Inc.
- Katie Haritos-Shea, Invited Expert
- Andy Heath, Invited Expert
- Takeshi Kurosawa, Mitsue-Links Co., Ltd.
- Dominic Mazzoni, Google, Inc.
- Edward O'Connor, Apple Inc.
- Janina Sajka, Chair, Invited Expert
- Joseph Scheuhammer, Invited Expert
- Richard Schwerdtfeger, IBM Corporation
- Jason White, Invited Expert

A.2 Contributors

The following individual(s) were previously active members of the working group or otherwise significant contributors.

Tab Atkins, Jesse Bunch, Chris Fleizach, Lachlan Hunt, Sangwhan Moon, Ryosuke Niwa, Rich Simpson.

A.3 Enabling funders

This publication has been funded in part with Federal funds from the U.S. Department of Education, National Institute on Disability and Rehabilitation Research (NIDRR) under contract number ED-OSE-10-C-0067. The content of this publication does not necessarily reflect the views or policies of the U.S. Department of Education, nor does mention of trade names, commercial products, or organizations imply endorsement by the U.S. Government.