test! sdfksfjtj Refactoring was originally conceived as a technique to improve softwares internal qualities -such as understandability and maintainability while preserving semantics.1 In prior work, we adapted the refactoring approach to improve a Web applications external attributes, such as usability.2 These Web refactorings consist of small navigation or interface transformations that enhance perceivable aspects of Web applications, such as user interaction and content presentation, while preserving functionality. Refactorings can also solve accessibility and usability problems for disabled users.3 Still, its usually impractical to address interface improvements for all audiences because disabilities can vary dramatically in nature (visual, cognitive, or motor), severity (blindness, color blindness, or strabismus) and extent (total or partial). In such different contexts, one for all is barely feasible.
1Web2Web Web3
When applying refactoring to improve internal qualities, developers decide which transformations to apply and where, because theyre the ones benefiting from the improvement. As Brian Foote and Joseph Yoder put it, Who better to resolve the forces impinging upon each design issue as it arises, as the person who is going to have to live with these decisions?4 Moreover, different developers might prefer alternative solutions for the same bad smell (that is, the design problem that motivated the refactoring1). Following on this general philosophy, we believe that end users should be able to tailor a websites interface for their own benefit.
Brian FooteJoseph Yoder41
We propose empowering users (or close representatives) with the ability to select, in their client browsers, their own interface refactorings for each site they access. We call our approach Client-Side Web Refactoring. CSWR allows for the automatic creation of different, personalized views of the same application to solve the particular bad smells that each user recognizes. (Developers should continue to focus on addressing general usability problems on the server-side, however,and reach the minimum level of accessibility,or A).
WebCSWRCSWRA
Here, we describe CSWR and a case study we ran with visually impaired users (though a similar solution could be applied for other disabilities).
CSWR Refactoring was originally conceived as a technique to improve softwares internal qualities -such as understandability and maintainability while preserving semantics.1 In prior work, we adapted the refactoring approach to improve a Web applications external attributes, such as usability.2 These Web refactorings consist of small navigation or interface transformations that enhance perceivable aspects of Web applications, such as user interaction and content presentation, while preserving functionality. Refactorings can also solve accessibility and usability problems for disabled users.3 Still, its usually impractical to address interface improvements for all audiences because disabilities can vary dramatically in nature (visual, cognitive, or motor), severity (blindness, color blindness, or strabismus) and extent (total or partial). In such different contexts, one for all is barely feasible.
1Web2Web Web3

When applying refactoring to improve internal qualities, developers decide which transformations to apply and where, because theyre the ones benefiting from the improvement. As Brian Foote and Joseph Yoder put it, Who better to resolve the forces impinging upon each design issue as it arises, as the person who is going to have to live with these decisions?4 Moreover, different developers might prefer alternative solutions for the same bad smell (that is, the design problem that motivated the refactoring1). Following on this general philosophy, we believe that end users should be able to tailor a websites interface for their own benefit.

Brian FooteJoseph Yoder41

We propose empowering users (or close representatives) with the ability to select, in their client browsers, their own interface refactorings for each site they access. We call our approach Client-Side Web Refactoring. CSWR allows for the automatic creation of different, personalized views of the same application to solve the particular bad smells that each user recognizes. (Developers should continue to focus on addressing general usability problems on the server-side, however,and reach the minimum level of accessibility,or A).

WebCSWRCSWRA

Here, we describe CSWR and a case study we ran with visually impaired users (though a similar solution could be applied for other disabilities).

CSWR Refactoring was originally conceived as a technique to improve softwares internal qualities -such as understandability and maintainability while preserving semantics.1 In prior work, we adapted the refactoring approach to improve a Web applications external attributes, such as usability.2 These Web refactorings consist of small navigation or interface transformations that enhance perceivable aspects of Web applications, such as user interaction and content presentation, while preserving functionality. Refactorings can also solve accessibility and usability problems for disabled users.3 Still, its usually impractical to address interface improvements for all audiences because disabilities can vary dramatically in nature (visual, cognitive, or motor), severity (blindness, color blindness, or strabismus) and extent (total or partial). In such different contexts, one for all is barely feasible.

1Web2Web Web3

When applying refactoring to improve internal qualities, developers decide which transformations to apply and where, because theyre the ones benefiting from the improvement. As Brian Foote and Joseph Yoder put it, Who better to resolve the forces impinging upon each design issue as it arises, as the person who is going to have to live with these decisions?4 Moreover, different developers might prefer alternative solutions for the same bad smell (that is, the design problem that motivated the refactoring1). Following on this general philosophy, we believe that end users should be able to tailor a websites interface for their own benefit.

Brian FooteJoseph Yoder41

We propose empowering users (or close representatives) with the ability to select, in their client browsers, their own interface refactorings for each site they access. We call our approach Client-Side Web Refactoring. CSWR allows for

the automatic creation of different, personalized views of the same application to solve the particular bad smells that each user recognizes. (Developers should continue to focus on addressing general usability problems on the server-side, however,and reach the minimum level of accessibility,or A).
WebCSWRCSWRA
Here, we describe CSWR and a case study we ran with visually impaired users (though a similar solution could be applied for other disabilities).
CSWR Refactoring was originally conceived as a technique to improve softwares internal qualities -such as understandability and maintainability while preserving semantics.1 In prior work, we adapted the refactoring approach to improve a Web applications external attributes, such as usability.2 These Web refactorings consist of small navigation or interface transformations that enhance perceivable aspects of Web applications, such as user interaction and content presentation, while preserving functionality. Refactorings can also solve accessibility and usability problems for disabled users.3 Still, its usually impractical to address interface improvements for all audiences because disabilities can vary dramatically in nature (visual, cognitive, or motor), severity (blindness, color blindness, or strabismus) and extent (total or partial). In such different contexts, one for all is barely feasible.
1Web2Web Web3
When applying refactoring to improve internal qualities, developers decide which transformations to apply and where, because theyre the ones benefiting from the improvement. As Brian Foote and Joseph Yoder put it, Who better to resolve the forces impinging upon each design issue as it arises, as the person who is going to have to live with these decisions?4 Moreover, different developers might prefer alternative solutions for the same bad smell (that is, the design problem that motivated the refactoring1). Following on this general philosophy, we believe that end users should be able to tailor a websites interface for their own benefit.
Brian FooteJoseph Yoder41
We propose empowering users (or close representatives) with the ability to select, in their client browsers, their own interface refactorings for each site they access. We call our approach Client-Side Web Refactoring. CSWR allows for the automatic creation of different, personalized views of the same application to solve the particular bad smells that each user recognizes. (Developers should continue to focus on addressing general usability problems on the server-side, however,and reach the minimum level of accessibility,or A).
WebCSWRCSWRA
Here, we describe CSWR and a case study we ran with visually impaired users (though a similar solution could be applied for other disabilities).
CSWRRefactoring was originally conceived as a technique to improve softwares internal qualities -such as understandability and maintainability while preserving semantics.1 In prior work, we adapted the refactoring approach to improve a Web applications external attributes, such as usability.2 These Web refactorings consist of small navigation or interface transformations that enhance perceivable aspects of Web applications, such as user interaction and content

presentation, while preserving functionality. Refactorings can also solve accessibility and usability problems for disabled users.3 Still, its usually impractical to address interface improvements for all audiences because disabilities can vary dramatically in nature (visual, cognitive, or motor), severity (blindness, color blindness, or strabismus) and extent (total or partial). In such different contexts, one for all is barely feasible.
1Web2Web Web3
When applying refactoring to improve internal qualities, developers decide which transformations to apply and where, because theyre the ones benefiting from the improvement. As Brian Foote and Joseph Yoder put it, Who better to resolve the forces impinging upon each design issue as it arises, as the person who is going to have to live with these decisions?4 Moreover, different developers might prefer alternative solutions for the same bad smell (that is, the design problem that motivated the refactoring1). Following on this general philosophy, we believe that end users should be able to tailor a websites interface for their own benefit.
Brian FooteJoseph Yoder41
We propose empowering users (or close representatives) with the ability to select, in their client browsers, their own interface refactorings for each site they access. We call our approach Client-Side Web Refactoring. CSWR allows for the automatic creation of different, personalized views of the same application to solve the particular bad smells that each user recognizes. (Developers should continue to focus on addressing general usability problems on the server-side, however,and reach the minimum level of accessibility,or A).
WebCSWRCSWRA
Here, we describe CSWR and a case study we ran with visually impaired users (though a similar solution could be applied for other disabilities).
CSWR