Advanced “Webble” Application Development Directly in the Browser by Utilizing the Full Power of Meme Media Customization and Event Management Capabilities

Micke Kuwahara, Yuzuru Tanaka
Meme Media Laboratory
Hokkaido University
Sapporo, Japan
mkuwahara@meme.hokudai.ac.jp, tanaka@meme.hokudai.ac.jp

Abstract— A meme media object, also known as a Webble, always come with a set of familiar generic behaviors together with another set of specialized ones for that particular Webble. But what if there is a need for a custom behavior or interface that was not originally intended when first created. With Webble technology, that does not need to be a problem. In this paper we will attempt to show how simple it is, due to the design and construction of Webbles, to insert new customizable behaviors in any Webble available, or control application level events and actions, all through an intuitive, user-friendly interface. We claim that within a few hours of combining generic Webble building blocks and the setting up of configurable event handlers directly in the web browser, without traditional programming, we can create any arbitrary Silverlight-based web application, ready to be shared to the cloud and the world.

Keywords- Meme Media Objects, Customize, Configure, Web, Share, Distribute, Resource, Interact, Participate, Federation

I. INTRODUCTION

Webbles are based on the philosophy of memes, as originally coined by Richard Dawkins [3][4], that every thought and knowledge shared by humans may collide with other thoughts and knowledge and then reproduce or mutate, all in favor of survival and adaptation. The meme is a paraphrase which is supposed to make us see that human knowledge and cultural expressions are like genes in the way they evolve.

A. IntelligentPad

The idea of the meme has stimulated the research on how to make human digital knowledge fit the meme description so that creativity may easier spread, evolve and enhance in a shared environment. One of these attempts was done by Tanaka and his group at Hokkaido University which gave us the theoretical as well as the practical creation of the IntelligentPad [1][6][7]. The purpose of this IntelligentPad, this meme media, is to work as smart containers of digital knowledge that may freely be connected and disconnected in arbitrary ways to form any imaginable compound object. Such an object can be anything digitally available in a computer today, from simple web blocks of images and textboxes to more complex widgets and even full featured applications.

B. Webble World

From that philosophy and research has now risen an evolved and extended web based framework that allows every user to bring their ideas and knowledge to the table directly inside most commonly used browsers or operative system in order to create a true, open, online federation of humanity's collected knowledge that is in constant progress and evolution. This framework is called Webble World [2].

C. The Webble

A Webble is the next-generation Meme Media object, available inside a web-browser. It can be developed by a programmer within ordinary software development restrictions. When it has been deployed to the web it can be downloaded into the browser via a specialized search engine by any Internet user together with other Webbles and combined together to form new compound Webbles which the user can configure so that they may solve some tasks, or present some content the user wants to share. Many such compound Webbles can also be combined as a larger and more complex Webble application. Webbles evolves step by step every time it gets reused and republished. Advanced web application development can be done, without additional programming, directly in the browser with these building blocks of ‘meme Lego’, called Webbles.

D. Background

As our understanding of culture and knowledge sharing gains, along with the evolvement of today's Web to Web 2.0 and the semantic Web 3.0, the need for new, more open and more powerful tools emerges, tools that fill our need to build and communicate, to be inspired and further inspire others, to develop and change, edit and contribute. Such a tool needs to allow all types of users, from any background and with any sets of skills. It needs to be open and free so it may evolve alongside the rich source of knowledge it is carrying. It should be easy to access from anywhere in the world and it should be fully adaptable.

We believe Webble World could be such a tool.

E. Webble Theory and Structure

Webble is the shortening of the acronym Web-Pebble, which in turn means 'Pad Enhanced Building Block Lifelike Entity' (on the Web). It aims to tell you that Webbles are
building blocks because they can fit together with any other Webble no matter what purpose or design that Webble have. They are also highly configurable and customizable and have the power to communicate with each other.

The Webble is, as explained before, constructed based on the IntelligentPad system with some enhancements, which in reality means, that a Webble is structured a certain way, has a wrapping or coating that encapsulate the internal unique code and implementation of a specific object, so that it will think of itself as a Webble and be able to do Webble things.

The Webble is divided in two parts, a Model and a combined View/Controller part, plainly referred to as the View (Fig. 1). These are two separate entities with similar structures but with different roles. The Model is considered to handle all internal matters that do not require any external interface, also known as the business logic, while the View deals mainly with all interaction with the user and holds all visual parts of a Webble. Any Model can be combined with any View at Webble design and development stage.

Furthermore, within the core of Webble and IntelligentPad design one will find the concept of slots, each which is defined inside the Model and the View. A slot is an externally available property parameter or method controller whose values may be viewed, exchanged, communicated and modified between present Webbles and also by users. The name slot tells us that we can see it as a hole or a plug where one may connect a contact in order to create a stream channel or path between two slots in two separate Webbles. This channel can be configured unilaterally or bilaterally.

Slot values can be of numerous types, from the classical numerals and strings, to more complex types as xml documents and object dictionaries. Slots can also be bounded to methods where the value of the slot can serve as method parameters. Slots can also be generated and bound to attributes of the visual objects in order to directly control the appearance of a Webble via the slot. Slots can be created, removed or reconfigured both at design time as well as at run time.

The internal slot values of a Webble can all be configured via platform provided configuration tools or forms and have their values changed by the user. There are no limits to how many slots a specific Webble may have or of which type its values can be.

Slot communication between Webbles requires that the Webbles are closely related, in order to make them aware of each other’s existence. This is achieved by structuring Webbles in hierarchical parent/child relationships. Webbles can have any number of children, but every child can only have one parent.

A Webble may then create direction controlled slot connections with its parent and its children, and those parent and children may in turn connect further in the relational chain. In most cases it is the task of the user to choose which slot in the parent and which slot in the child should communicate with each other, but there are cases where advanced Webbles have the ability to setup such connections on its own. As soon as such a communication channel is configured, the Webble will handle the communication on its own by three ways of control methods. Whenever the parent have any value change in any slot it will fire the 'Update' message, informing every child that something have changed; it is then the task of the child to use the 'Gimme' message in order to retrieve the value of any specific slot from the parent to see if the update concerned them and the value of the slot they care for has changed. After that, it is then within the scope of the child to react upon the value collected. If instead it is the child slot that is altered it will transfer that alteration over to the parent with the 'Set' message to the connected parent slot, which in turn may make the parent react on the value change. Internally, both in the View and the Model, all slot changes fires a slot reaction method which sometimes does nothing and in other cases do a lot, maybe even start changing other slot values. A similar structure of communication is going on between the Model and the View within the Webble itself.

In 'normal' slot communication a Webble can only have one slot channel open with each relative (parent or children) at one time. But there are, as we will show further on, of course ways to set up multiple channels by using for example a specialized primitive Webble like the Event Action Manager or the Slot Subscription Manager, which can handle that sort of structure. Fig. 2 below shows a message flow diagram of the above described communication of slots between two related Webbles.
A Webble can be duplicated freely, either as a separate entity or by sharing the Model with its original. In the latter case, a duplicate is called a shared copy. This is another form of internal communication between Webbles, which does not require any parent/child relationship and not only affects the slot values but the Model as a whole.

A Webble is not only defined as described above, with programming code. An even more important part of the Webble is the Webble definition or configuration file which is described in XML and hosts all internal values and properties of a particular Webble, like slot values, children, connections, model and much more depending on Webble class. It is this XML file that separates primitive Webbles (code generated only) from compound ones.

There are two levels of Webble construction, one is the creation of primitive Webbles which means to create some useful features in code and wrap them into a Webble, to then publish its executable online. The other level is the configuration of primitive Webbles, which under the hood means to create an XML definition file, but for the user means attaching and configuring Webbles together inside the browser environment with mainly basic mouse operations and then save the result online. In both cases the goal is to make the creation available for the public.

One of the major strengths of the Webble Meme Media Object is that though recognizable in structure and design as well as in human interaction interfaces, they are never limited to these only. A Webble can look, feel and behave any possible way, only limited by the imagination of the developer, but it will in the end always be what we expect it to be; just a Webble.

F. Technology and Access

The rendering technology for the Webble is Microsoft Silverlight web application framework, mainly programmed in C#, XAML, HTML and JavaScript.

The simple, traditional way in using Webbles is by visiting a Webble World Web site (Fig. 3) (e.g. Google “Webble World”), where one may download readymade Webbles to use, create one's own compounds or applications or download the developers pack to create one's own primitives.

Figure 3. Some Webbles loaded into the browser.

II. Making a Game

It is by keywords like available, customizable, and combinable that one is supposed to describe the Webbles existence, and in this paper we hope to show the validity in the use of those words by closer examine a few Webbles like for example the Event/Action Manager and also lightly study the process on how to create a traditional solitaire card game directly in the browser without any game specific components.

A. Problem Description

Even though there are many generic customizations available within the generic Webble, there is still probably more often when an imaginative, creative user looks for a behavior or customization that is not within the range of the Webble in question, not generic, nor specific.

A common approach to solve such a problem is to think in terms of developing a new Webble that more or less do exactly the same thing that the first Webble did but with the added required functionality. In most cases, even though common, that approach is a bad one that will flood the Webble repository with innumerable primitive Webbles of similar kind. A better approach, which Webbles fully support, is instead to create a behavior Webble that will add, alter and/or manipulate any Webble it is attached to.

The ultimate approach is of course still under some debate and scholars of non-programming, mash-up developments and end user interfaces [8][9][10][11][12][13] might definitely want to add to the discussion, but since Webble World is one of few actually fully working and available systems out there and in contrast to for example ‘Scratch’ by MIT Media Lab that requires the installation of a desktop application to work, Webble World runs completely online in the web browser. Webble World is also not mainly focusing on mash-up like behavior and animation capabilities, which other systems seem to do, but also allows much more advanced development which will be seen later in this paper.

B. Loose Coupling

Since All Webbles and even its internal parts are very loosely coupled to both the platform in which it resides but also to other related Webbles and its separate internal parts as Models and Views, it is natural to see that every single piece of behavior, design, appearance, structure, interface not only can, but most likely should be separate parts as well, which when combined create a fully functional and useful widget or application. Similarly this means that every widget can easily be detached into smaller pieces which can be replaced or reused in different scenarios. A Webble is therefore much more than just an object with a predefined singular purpose, it can also be a tool, or a device which, when attached to other Webbles allow us to interact differently to its parts.

C. The Event-Action Matrix approach

A Webble developed for the purpose to redefine other Webbles and allow customizations of Webble behavior is the Event Action Manager (EAM) Webble known as the Event-
Action Matrix. Its name comes from how the user interacts with the Webbles interface in order to control and configure Webble behavior both internally as well as externally on application level, as can partly be seen in Fig. 4.

Through the interface provided by the EAM Webble the user can set up events (triggers) that will cause an action to fire. These events are created and located on the leftmost column in a matrix table where each following column are representing all other related Webbles connected to the EAM Webble as well as the EAM Webble itself and the Webble World Platform. The actions will be found available in the created cross section of a Webble (or platform) column and an event row. As the event is defined an action can then be configured in the proper cross section cell in the matrix, below the targeted meme media object, which will be initiated by the triggering of the event to the left and which will affect the Webble found on the top. The interface is simple and straightforward to use by almost any user, with a basic understanding of concepts such as ‘Mouse Down’, ‘Slot Value Change’ etc which are easily learned via the online available Webble manual. But for more experienced users the EAM Webble also allows the creation of much more complex actions with the help of JavaScript parsing. That would of course imply some traditional programming, but it is only an option for more advanced cases and is never required or necessary.

The core events and actions available are currently predefined inside the EAM Webble, and are not possible today to control from the browser, meaning that the user cannot add a completely new event or action to the Webble without writing code. Having said that, most known common events and actions related to Webbles and RIA systems in a browser such as for example traditional mouse and keyboard events have been included, and they are all configurable in several ways to fit the need of most users.

We think the best aspect and primary strong point of the EAM Webble is of course its matrix layout that easily and fast display to the user what events, by what Webble or user action, causes what actions, to what Webble or the system as a whole. What is worth pinpointing even further is the fact that the EAM Webble is exactly just that, a Webble, meaning that it is not the system or platform itself that provides these new functionalities and ways to configure the Webble environment but only another Webble. Which in turn indicates that the possibilities for Webble development and future Webble repositories are, if not limitless, at least capable to always be at the forefront of what current available web technology allows us to do.

D. The Interaction Tweaker

Another Webble, whose purpose is to enhance any other existing Webble, though not used in the solitaire game example further on, is the Interaction Tweaker Webble that when attached as a child allows us to modify the parent’s interaction interfaces such as the Webble menu and interaction ball graspers in its border. Similar to the EAM Webble also this Webble allows complex scripting alongside with, according to the authors, an intuitive, straightforward, easy to use interface (Fig. 5).

E. The Idiot

The Idiot is the name of a popular solitaire card game (Fig. 6 & Fig. 7) that we for the purpose of this paper quickly threw together to strengthen our claim that it is both fun, easy and possible to create any arbitrary web application through plain Webble interaction of only simple generic Webbles and in-browser configurations.

Besides the EAM Webble there were only five other Webbles used; Image-, Textbox-, Select List-, Window- and Text Label Webble. All card piles were created by List Webbles. For selections and click memories could have been used simple slots, but for clarity and easy overview textboxes were used instead. All text that was going to be displayed to the user was mainly done so with Label Webbles.

Most Webbles were directly connected as children to the EAM Webble, but that could have been arranged differently if needed or required. Some of the display (Label) Webbles were connected to those Webbles which held the slot whose value we needed to display and some of the stand alone Webbles like the game title was not related at all, which of course ‘forced’ us to save the Webble as an application rather than a standalone compound Webble.

Figure 4. This is the inside of an Event-Action configuration.

Figure 5. Tweaking of a Webble interface in action.

Figure 6. Most Webbles needed were only running the background logic and was later hidden to the final user.
The initial threshold while creating the idiot solitaire was an unexpected one. Since we are software programmers and system developers at the foundation we have a certain way to think and approach a problem which we discovered complicated things when we were working with Webbles. ‘For’ loops, regular ‘if’ statements, function calls etc was not going to help us much, we instead had to learn to think a bit more like a non-programmer to realize how we best could construct the logic of the game.

Most of the game logic ended up in the EAM Webble where mouse click events were recorded, but also user hidden intermediary events was setup to catch changes in game status and create follow up events, i.e. the first click fires the placing of the first card which then fires another event to place the second etc. In total, 25 events with one or more consecutive actions were created to control the full game and basic user actions. Some events used JavaScript for enhanced complexity as which can also be seen in Fig. 7 below, but with future increase of available generic and specific Webbles that will be less and less needed.

We found that the EAM Webble user interface is simple and straightforward, but there are improvements that can undoubtedly be done which we noticed as we used it more extensively than before, like for example the use of, hard to read, Webble ids instead of, for example, images or more user-friendly names. Also the matrix scrolling requires more sophistication, so that relations between certain events and actions becomes much more obvious, when navigating in a large matrix.

It took about 6 hours to build the game, which maybe is not that impressive to an experienced programmer working in traditional tools as e.g. Visual Studio, who could maybe do it in similar time or even slightly less if putting the mind to it, but considering it was done directly in the browser without traditional programming or basically no writing of actual code or having to compile or distribute anything it is definitely a "first" attempt in the right direction. We also believe that with some user interface improvements and more experience in meme media object thinking, in contrast to classical programming, we strongly have the feeling that the time of development can be highly reduced. Not to mention a future where more generic Webbles are available in the common repository, which will definitely simplify and speed up development.

### F. Evaluation

With these Webbles and similar to come there are no limitations left to what a custom Internet visitor can construct and create inside his/her own browser without practically any form of regular programming, but instead mainly by basic mouse operations.

And through the Webble platform, with only a mouse click, one can make such creations available for the rest of the world to use and interact with, not to mention be inspired by, to modify or add to.

### III. OTHER WEBBLE PROJECTS

But games and minor amusements is far from the only thing that Webble development allows, we therefore would also like to have a quick glance into some other on-going Webble projects and applications being currently developed, in order to further strengthen our argument for meme media development as a valid concept.

#### A. Trial Outline Builder

This Webble application named Trial Outline Builder (TOB) aims to simplify the construction of a clinical trial, as well as the collection of data on patients being treated and also support researchers in analyzing the trial.

Other clinical trial management software tools are often no more than sophisticated document handlers, where data is difficult to visualize and often leaves the researcher on his/her own, with weak workflows and seldom comparable studies. The integration of Webbles has resulted in a digital interactive environment where a workflow scheme of medical events and treatment elements can easily be structured and manipulated, like pieces of puzzle or Lego. Not only does this give a better overview of the treatment plan of the trial but it also allows the user to enter, extract and manipulate data in a more intuitive way through the whole life cycle of the trial management.

![Figure 8. The game as it looks in action for the user.](image)

![Figure 9. The Analyzing View of TOB where each individual object is a Webble](image)
B. Assets

The Assets application allow the user to search a vast database of museum artifacts provided by the Europeana group and create new document structures and relationships between these objects together with user generated content.

![Image](https://example.com/asset_image.png)

Figure 10. A Screen shot of the Assets application running.

C. Solar Bike

In Germany has been developed a virtual lab environment where each Webble represent a physical object that when put together properly creates a mechanical biker driven by a virtual sun. The same institute has also developed e-learning tools in Webbles for University math and science courses.

![Image](https://example.com/solar_bike_image.png)

Figure 11. The finished result after constructing the biker.

D. Other Applications

Data analyzing and statistical tools, maps and visualization apps, data retrieval and conversion Webbles and much more are also available for usage and app development. The Webble repository increases constantly and steadily and will continue to do so until we have all the bricks we need to build anything we can imagine.

IV. CONCLUSION

In order to simplify participation, to enhance the options of collaboration and to allow both re-usage and re-combinations of web resources for both skilled and ordinary web users, there has been developed a framework called Webble World where users can publish, configure, combine and save a multitude of independent modules that wrap both contents and functionalities.

We have developed behavior Webbles for that framework and used them to create a, if not useful, at least an amusing web application directly in the browser to prove the concept and theories of meme media object development.

We believe that with the continuous evolution of existing and future Webbles and through processes similar to natural selection Webble World or any future counterpart based on similar theories will be a force to reckon with when it comes to future software development in a cloud based online environment where re-inventing the wheel will be less and less appealing and necessary.

In the light of Tangible Edutainment Media for Playful Evolution of Knowledge and Understanding, especially in regard of its sub topics, we find that Webbles as interactive media blocks allows multiple forms of human-computer interaction in an immersive environment and with its highly visual interface inspires development in game-like learning situation. Within this environment of free construction and shared repositories of knowledge and applications it is not difficult to realize the potential for constructivist learning through direct manipulation and the evolution of development. Not only will users learn from developing multimedia interaction environments, but other users will further that knowledge caused by the developed applications purpose to teach, and as technology advances and the underlying framework mature by multiple channels of quality assessment and an increasing pool of Webble resources we believe Webble World may be the multi-use tool of World Wide Web Interaction.

REFERENCES


