

# Child Development <sup>101</sup> for the Developers of Interactive Media

An Overview of Influential Theories of  
Child Development



This is a handout packet for a workshop for the designers of children's interactive media. Designing IM that is successful depends on a variety of factors, including a solid knowledge of the capabilities and learning patterns of the intended users (children).

This packet provides an overview of what children can do at each age, along with brief summaries of major child development and learning theories. Also included are essays on what makes a particular piece of software better than others. We have attempted to write concisely, in a style that is free of educational jargon.

Keeping in mind some key child development trends early in the software conceptualization process can greatly increase the likelihood of a child wanting to spend time with your product.

---

First edition, 1996. Revised January 6, 2006  
by Ellen Wolock, Ed.D, Ann Orr, Ed.D. and Warren Buckleitner, Ph.D.  
© 2006 Active Learning Associates.

All rights reserved.

Reproduction in part or in whole without written permission is strictly prohibited.  
Printed in the United States of America.

**Picky Teacher** and **Children's Technology Review** are trademarks of  
Active Learning Associates, Inc. 120 Main Street, Flemington, NJ 08822

Internet image references: 16mm projector -- ([www.xs4all.nl/~wichm/cinimage.html](http://www.xs4all.nl/~wichm/cinimage.html)), flashlight ([cartalk.cars.com/Store/Car-Kit/list.html](http://cartalk.cars.com/Store/Car-Kit/list.html)), slide projector ([www.gpc.peachnet.edu/~ecoway/equipinfo.htm](http://www.gpc.peachnet.edu/~ecoway/equipinfo.htm)), Jean Piaget ([www.piaget.org](http://www.piaget.org)). Other images from Studio 1 Photography in Ann Arbor, MI.



# Contents

## **Section 1. How Children Learn .....3**

Intrinsic Motivation Factors: I WANT to play! .....	3
Piaget: Constructivism, and his Theory of Cognitive Development .....	4
Other Influential Theories .....	8
Behaviorism .....	8
Constructivism .....	8
Social Learning Theory .....	8
Social Constructivism .....	8
Information Processing Theory .....	7
Bloom’s Taxonomy.....	7

## **Section 2. Developmental Stages .....8**

Birth to 18 Months.....	8
18 Months to 2 1/2 Years .....	8
2 1/2 to 3 Years.....	8
3 to 4 Years.....	9
4 to 5 Years.....	9
5 to 6 Years.....	9

## **Section 3. Specific Tips .....10**

Capturing the Magic of Interactive Media .....	10
Enemies of Magic—(What Makes a Product “Dust”) .....	11
What All Software for Children Must Have .....	12
A Dirty Little Secret: Many Kids’ Console Games are Too Hard .....	12
Effects of Praise and Reinforcements on Engagement .....	13
Eight Lessons for Structured Activities .....	14
Mouse Use Patterns .....	14
Self Evaluating Your Product .....	17

# Section 1. How Children Learn

## Understanding Intrinsic Motivation



The efficiency of children's learning is increased when they have a stake in the task. It is theorized that intrinsic motivation plays a critical role in the degree to which a child will become engaged with an activity. Several factors have been identified by motivation theorists (Weiner, 1986, White, 1959; Maehr, 1983; Stipek, 1986; Lepper, 1973 to name a few) as being central to the development of intrinsic motivation attributes in a learner's behavior. These factors can be used as a framework for understanding a child's actions when he or she is using software or for assessing the overall quality of the software.

### 1. Enjoyment — Children choose activities that they like to do, and avoid activities that are frustrating, static or boring.

#### Implications for software design:

First of all, when designing an activity, make sure the child finds initial success within the first 10 to 20 seconds of play. Sign-in screens should be intuitive; the first activities easy and fun. Nothing kills enjoyment faster than failure. Secondly, don't underestimate kids. Some designers think that children don't pay attention to fine details because they are too young ("don't worry ... these are just little kids"). Put yourself in the child's shoes— would you want to play the activity you are designing? Walt Disney understood that the process of creating a successful children's film is just as difficult as that of creating a film for adults. Disney films appeal to all ages, stand the test of time and are watched over and over again. Similarly, the success of The Living Books and the Humongous programs can be tied in part to their clever use of animation and humor, which aren't used randomly or without purpose. When software is easy to use and respectful of children, kids are more likely to enjoy it and use it.

### 2. Control — Children avoid activities in which they have no control. Good software increases children's feelings of control by providing an environment where their actions have impact.



#### Implications for software design:

First, good software sends instant, snappy control messages (such as an action or audio cue) with each mouse click. A crisp, responsive interface increases feelings of control. Avoid trying so hard to grab the child's attention with music, video or animation that the program's responsiveness suffers. Second, good software allows many opportunities for child input. Why are programs like Kid Pix such favorites? Because every action the child makes results in something happening on screen. The child is leading the way, not the software. Third, good software also always leaves an intuitive "back door" or "go back" icon that is in a consistent place on every screen. Once a child understands that he or she can reverse a choice or decision, they are more likely to explore further or try a harder challenge.

### 3. Interest — Children are more likely to engage in an activity when their interest has been sparked.

#### Implications for software design:

First, remember that the adage "variety is the spice of life" holds true for software design. Make sure that each play offers something new or incorporates open-ended elements. Children love surprises and novelty. Don't make a "one time through and we're done" program. Second, get to know as many real live kids as you can. What are they interested in? Fads come and go, but kids always want software with characters they can relate to, good story lines, quality music, humor, and familiar items and themes. Programs about animals, for instance, fly off our review shelves. Likewise, adventure programs are always appealing.

### 4. Feelings of Competence — Children develop feelings of competence if they think they have a reasonable chance of success.

#### Implications for software design:

It is the developer's responsibility to provide children with materials and activities that are at or near their developmental level. This, of course, refers to program content, but also to its design as well. Make sure you understand the notion of "minimum user competency." In other words, the challenge should be in the activity itself, not in the physical operation of the program (non-intuitive icons, reading required when the target audience is preschool, etc.) Another important concept to understand is that of "motivation inertia." In other words, make sure your software includes elements that build on previous success, allowing greater challenge that is tailored to the child's abilities. Appropriate pacing and leveling of activities is critical— too fast, and the child builds a failure bank rather than competence bank. Too slow, and the intrinsic motivation wanes. Regarding the latter scenario, if extrinsic reinforcers are used such as "nice job" or "try again," make sure they don't slow the pace of the activity (especially when there is a timed element). Teachers have learned that one of the most effective punishments is "time out", because children hate to wait, yet that's exactly what some software design does.

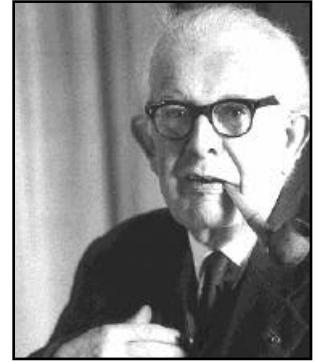
Memorize these 4 factors



# Piaget

## Constructivism, and his Theory of Cognitive Development

The research and writings of Jean Piaget (1896-1980), the Swiss psychologist, have had an enormous impact on the field of cognitive development. Piaget's observations and theories have helped us better understand the way children think and learn. Piaget's theories can be divided into two parts: his description of a set of discreet stages through which children proceed, and his explanation of a set of processes that help move a child from one stage to the next. A basic understanding of Piaget's theories can help the software designer create more developmentally appropriate software. Note that students of educational psychology have successfully refuted some of Piagetian theory. However, the big ideas are valuable and commonly used in the design of educational activities.



Learn more about Piaget, his life and his work at [www.piaget.org](http://www.piaget.org). A good book on Piaget for novice readers is *The Piaget Primer: Thinking, Learning, Teaching* by Ed Labinowicz (Menlo Park, CA: Addison-Wesley Publishing Co., 1980).

### Sensorimotor, birth to 2 1/2 years. (a period of sensory input and physical actions)

Piaget described the ages of 0 to 2 years as the Sensorimotor Stage. All learning is done via physical exploration of the environment. As the child interacts with people and things, pleasing reactions are eventually noted, making the action more likely to be repeated. In the later portion of this period, the child begins to actively experiment, trying out various actions and reactions in a more purposeful manner. By the end of this period, the child has acquired an initial set of concepts dealing with space, objects and causality.



Sensorimotor children think like a flashlight—where the beam shines is where they think. The rest of the world doesn't exist.



- ✓ Babies initially think that objects out of sight aren't there, but later understand that the object doesn't really disappear (like in peek-a-boo).
- ✓ Children learn through the direct manipulation of objects, using all senses (touch, taste, sound)
- ✓ Children learn through the repetition of actions and imitation.
- ✓ Children understand simple cause and effect.

### Preoperational, ages 2 1/2-7. (a period of representational, prelogical thought)

From approximately age 2 to age 7, Piaget described the child as being in the Preoperational Stage. Language acquisition is a major goal, as is "object permanence" or the idea that objects continue to exist even when they are out of sight. This is the first building block of memory and higher order thinking skills. Piaget also believed that children at this age fail to understand that the mass of an object is unchanged even when something is done to it. For example, if you take a short glass of milk and pour it into a taller, narrower glass, children in the Preoperational Stage will think that the taller glass contains more milk.



Preoperational children think like a slide projector. They can typically hold distinct thoughts in mind, but have trouble mixing them, or understanding that they can affect one another.

- ✓ Children begin to represent experiences through play and communications.
- ✓ Children are generally egocentric, less able to take another's perspective.
- ✓ Children consider the current condition of what they see. For instance, a small banana cut into lots of little pieces is "more" than a big banana cut into just a few pieces.
- ✓ In the early period of this stage, expressions may be taken literally, e.g. keep an eye on the ball.



## Concrete-Operational Stage, ages 7-12.

(a period of focused logical thought)

From about 7 to 12 years, the child is described as being in the Concrete-Operational Stage, a period characterized by a more mature understanding of the world and objects around them. They understand that you can do things that change appearance of an object, without changing the essence of the object. For instance, children at this age understand that a certain amount of liquid has the same volume regardless of how it looks, or a ball of clay has the same mass even after you smash it into a patty. This kind of thinking forms the basis for scientific exploration and thought (kids at this age love science), but they still rely upon concrete objects and experiments to form their ideas.



Concrete operational kids think like a motion picture projector; a step up from distinct "slide by slide" processing. But they still have trouble jumping, or mentally juggling different ideas. This is why they rely on concrete materials.

- ✓ Children are tied to their direct experiences, but can consider and coordinate more than one dimension.
- ✓ Children understand time, space and number. Children can conserve, understanding that objects are the same if their original state was equal, e.g. a small banana is smaller than a large banana, no matter how it is sliced.
- ✓ Children can take another's perspective.
- ✓ Children in this period still learn best through concrete experiences.

## The Formal-Operational Stage, ages 12-17

(a period of unlimited, logical thought)

Formal-Operational thinking is said to begin around age 12. Here, the adolescent begins to use abstract logic, and no longer relies on concrete objects to form his thinking. Learning can occur through verbal reasoning, and by taking the perspective of others. Pre-teens/teens in this period formulate their own hypotheses about causes and solutions. They are now able to rely on abstract symbols to learn.

- ✓ Preteens/teens can rely on symbols to understand and learn.
- ✓ Preteens/teens understand complex concepts like density.
- ✓ Egocentrism may disappear completely with the capacity to think and reason beyond own beliefs.
- ✓ A sense of fairness and equality supersedes adult authority.



Like a computer controlled DVD player, formal operational thinking children can quickly skip from one idea to another, hold multiple contrasting ideas in memory, and evaluate the relevance of different ideas.

## Defining Constructivism

How do children move from one stage to the next? Piaget argued that children "build" knowledge via a cycle of repeated and expanded interactions with their environment. Piaget described this process as having two mechanisms. One, he referred to as **assimilation**. When encountering something new in our environment (which is the prerequisite to learning), humans first try to incorporate that new thing into our existing mental framework.

**Accommodation** is the complement to assimilation. Accommodation occurs when we have to adjust our existing mental framework in order to make room for that new "thing". Piaget said that these two processes are occurring all the time, back and forth as we experience (learn) new things in our environment. When children encounter something new, they are slightly off balance, experiencing what Piaget called **disequilibrium**. The child naturally seeks **equilibrium**, or a balance between interacting factors inside and outside the child.

Real world examples are easy to find. Learning a difficult sport, such as golf (trying to sink a putt, or hit a drive down a fairway), water skiing or snowboarding all have definable moments of assimilation, accommodation and equilibrium.

**Software developers can use the tenets of Piagetian theory when they design new activities and programs. How? By gently nudging the child from equilibrium to disequilibrium to equilibrium again.**



# Other Influential Theories

## Behaviorism

This theory asserts that behavior can be explained entirely in terms of observable responses to environmental stimuli. Influenced by the conditioned-reflex experiments of Pavlov, behaviorism was introduced in 1913 by J.B. Watson, who, denying both the value of introspection and the concept of consciousness, emphasized stimulus-response laboratory techniques. B.F. Skinner concerned himself exclusively with the relationship of observable responses to stimuli and rewards, and one result was the concept of mastery learning, which was applied in the 1950's as "teaching machines". Edward Thorndike was another important proponent of behaviorism; his work looked at the role of rewards and consequences and the technique of breaking tasks into small parts to be learned. **Software that is very linear and scripted or that relies heavily upon external rewards draws from the behaviorist perspective.**

## Constructivism

This school of psychology asserts that children actively construct their own knowledge from prior experiences — a process of fitting together new experiences with old to create a new reality. The theories of Jean Piaget (1896-1980) have been used to support constructivist curricula which include the open classroom movement, whole language, and others. The idea that the child is an active, not passive, learner is key to this theory. **Software that is child-led using open-ended components or virtual manipulatives draws from the constructivist theory.**

## Social Learning Theory

The work of Albert Bandura (b. 1925) gave rise to the social learning theory. Bandura emphasized the social aspects of learning, for instance, the importance of observing and modeling the behaviors, attitudes and reactions of others. In other words, Bandura claims that much of what we learn is attained by watching other people. Bandura sees learning as a continual, reciprocal interaction between cognition, behavior and environmental influences. The learner's attention, memory and motivation are seen as key determinants of learning. **Software that models desired responses or that provides children with opportunities to see other kids learning and doing can be said to draw from social learning theory.**

## Social Constructivism

Lev Vygotsky, a Russian psychologist and philosopher in the 1930's, is most often associated with the social constructivist theory. He emphasized the influences of cultural and social contexts in learning and supported a discovery model of learning. Vygotsky believed that learning and development is a social and collaborative activity that cannot be "taught" to anyone. It is up to the student to construct his or her own understanding in his or her own mind, while the teacher acts as a facilitator. Vygotsky maintained that learning should take place in meaningful cultural contexts. **Simulation programs like SimCity are perfect examples of social constructivism, as are online games which facilitate the communication between two or more players.** Vygotsky's concept of the "zone of proximal development" is a useful idea for software developers. This "zone" has been defined as the distance between a child's independent problem solving and his capabilities of problem solving while under adult guidance or the guidance of more capable peers. The "zone" is where you want to be when teaching a child— just slightly beyond what he can already do by himself. The same goes for software activities, you want them to be challenging, but



Lev Semyonovitch Vygotsky was born in 1896 in Byelorussia (Soviet Union). He began his career as a psychologist in 1917 and only pursued this career for 17 years before his death from tuberculosis in 1934.

## Key Definitions

**intermittent reinforcement**— a type of reinforcement schedule in which the praise or reward is given only once in a while (like a slot machine). Of all the reinforcers, this one is the most powerful when used in software.

**minimum user competency**— the lowest level (entry level) skill a child must possess in order to be successful with an activity. A menu that requires reading raises the MUC, for example.


**responsivity**— one of the variables considered to be most related to engagement. An example of responsivity in software is an immediate response to a mouse click or key stroke.

**zone of proximal development**— or "ZPD" is an especially useful idea for software developers. This "zone" has been defined as the distance between a child's independent problem solving and his capabilities of problem solving while under adult guidance or the guidance of more capable peers. The "zone" is where you want to be when teaching a child— just slightly beyond what he can already do by himself. A "smart" character that suggests a new activity based on previous performance is one example.

not overwhelming, and you want to give the child enough support while doing the task that he succeeds at learning something new. Programs that track a child's past performance and automatically offer slightly more challenging activities are using the concept of "zone of proximal development".

## Information Processing Theory

Based on the work of George Miller and others, the Information Processing theory of learning maintains that children are actively processing, storing and retrieving information (much like a computer) and that teaching involves helping learners to develop information processing skills and apply them systematically to mastering the curriculum. Two major principles of this theory are that short term memory (or attention span) is limited to seven chunks of information and that processing information in sequential steps is a fundamental cognitive process. CAI (computer assisted instruction) software uses these principles. Tasks are broken into sequential steps, connections between new and old information are highlighted, retention strategies are suggested and there is ample opportunity for repetition and review of information. **Developers of software designed to teach memorization of facts, reading, etc. should explore this theory further as its concepts can be easily integrated into learning games and activities.**



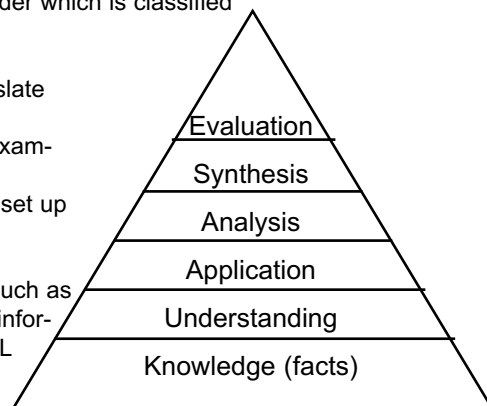
Now you know why teachers pull in the big \$\$

## Bloom's Taxonomy

In 1956, a group of educational psychologists led by Benjamin Bloom found that over 95% of the test questions students encounter require them to think only at the lowest possible level...the recall of information. Bloom identified six levels within the cognitive domain, from the simple recall or recognition of facts, at the lowest level, through increasingly more complex and abstract mental levels, to the highest order which is classified as evaluation.

1. **Knowledge:** define, list, match, order, name, repeat, memorize, recall
2. **Comprehension:** describe, sort, classify, report, review, identify, review, translate
3. **Application:** demonstrate, illustrate, solve, employ, use
4. **Analysis:** categorize, appraise, calculate, compare, contrast, distinguish, examine, question, test
5. **Synthesis:** compare, formulate, manage, organize, plan, prepare, propose, set up
6. **Evaluation:** assess, defend, estimate, judge, predict, support, value, test

Software has much to contribute to "higher order" thinking, through simulations such as **Sim City** or **Oregon Trail**, where children must continually evaluate and synthesize information related to a long term task. Other activities such as programming in HTML or creating with a database or spreadsheet are also examples of these skills.



\* Adapted from: Bloom, B.S. (Ed.) (1956) Taxonomy of educational objectives: The classification of educational goals: Handbook I, cognitive domain. New York ; Toronto: Longmans, Green.

# Section 2 Developmental Stages



The following pages offer descriptions of children's "ages and stages", and what children can do on the computer at what age. Also presented are some general developmental milestones that children display during their early years. These age estimates and developmental attainments are based on several instruments used to track young children's growth. While this compilation will help you get a sense of what children can do at each age level, keep in mind that individual children acquire skills at different rates.



Knowledge of children's developmental abilities is particularly important when designing software for younger children (below age 6). Many aspects of software will be affected, such as menu design, content, reinforcement messages and so on.

## Birth to 18 Months

To babies, the computer is little more than a giant busy box. They love to look at the colors on the screen, hear the sounds from the speakers, mouth the mouse cord and touch the keyboard. Don't expect young babies to make the connection between their movements of the mouse or keyboard and events on the screen.

## 18 Months to 2 1/2 Years

While the computer is still viewed as an electronic busy box, at about 18 months, children first begin noticing that they can have an effect on objects on the screen. They'll need some assistance, as they don't have the necessary skills to use the mouse. We've seen some families successfully use a touch screen with children in this age group. Touch screens aren't really necessary though, as programs like Reader Rabbit's Toddler will let children press any key on the keyboard to hear a favorite song or discover pictures and animations.

- ✓ Can recognize pictures of objects.
- ✓ Can identify body parts on self or on a doll.
- ✓ Can place individual shapes on "form board" type puzzles.

- ✓ Can use a pencil to imitate a vertical line.
- ✓ Can match objects by color.
- ✓ Can match objects by simple shapes.
- ✓ Can understand the concept of "here".
- ✓ Can remember a missing object if it is presented and then taken away.
- ✓ Begins to categorize objects according to function (e.g. places all of the spoons together).
- ✓ Enjoys and remembers nursery rhymes.
- ✓ Enjoys taking things apart and putting them together again.
- ✓ Has limited attention span.
- ✓ Can name 1 to 2 colors.
- ✓ Enjoys copying activities of parents and siblings.
- ✓ Generally plays along side of peers rather than cooperatively with peers.

## 2 1/2 to 3 Years

Age 2 1/2 for many children is a real turning point when it comes to computer use. Not only can they sit for a bit longer (we've seen kids who can sit for as long as 1 hour), but many have the fine motor control to use a mouse independently. Now they can easily negotiate the activities in programs like Reader Rabbit's

Toddler, or Fisher-Price Ready for School: Toddler. They especially love singing along with the music, while watching events on the screen. Keep in mind that children develop their mouse skills at different rates. Kids may be ready earlier than age 2 1/2 or they may need more time. Also remember that computer use is a very social activity for young children ... they love sitting in a parent's lap to experience the activities together.

- ✓ Can describe the functions of objects (e.g. "What do you sleep on?").
- ✓ Asks "why" and "how" questions.
- ✓ Can anticipate consequences and understand the impact his or her own actions can have (e.g. understand the relationship between clicks of mouse and actions on the screen).
- ✓ Can recognize several colors.
- ✓ Knows the sounds that animals make.
- ✓ Can count to 2.
- ✓ Engages in simple fantasy play (driving vehicles, cooking meals, feeding baby, etc.)
- ✓ Is able to answer simple questions.
- ✓ Usually speaks in short but complete

These guidelines are no substitute for time on a playground.

sentences.

- ✓ Understands the concept of "now".



### 3 to 5 Years

These are the first real years of independent computer use. Children can now manipulate the mouse expertly (providing they've had plenty of time to practice), and can use a variety of programs.

Kids at this age typically want to share the fun with a friend. Electronic storybooks work especially well at this stage, as do simple adventure programs like the Putt-Putt series from Humongous. The preschool activity packs like Millie and Bailey's Preschool, and creativity programs that let them print their work, are also good for this age group.

#### Age 3 to 4 Milestones

- ✓ Recognizes most colors.
- ✓ Can identify simple shapes (e.g. square, circle, triangle).
- ✓ Understands the concepts of "same" and "different".
- ✓ Can play independently for extended periods (approximately 20 minutes).
- ✓ Begins to play cooperatively with peers.
- ✓ Enjoys and remembers a favorite song.
- ✓ Can follow two simple directions in the correct sequence.
- ✓ Can complete a 4 piece puzzle.
- ✓ Can copy a cross (+).
- ✓ Can draw a circle.
- ✓ Build towers of 10 or more blocks, and can build simple bridges.
- ✓ Can recognize many letters.
- ✓ Counts to ten.
- ✓ Shows some understanding of one-to-one correspondence (when counting, each number represents an object being counted).
- ✓ By age 4, can use a pair of child-size scissors to cut on straight, thick lines.

#### Age 4 to 5 Milestones

- ✓ Understands the concept of "today".
- ✓ Makes fine size discriminations (e.g. can order objects according to size, can match objects according to length).
- ✓ Makes broad classifications according to type (e.g. animals, foods, clothing).
- ✓ Understands the sequencing of events (e.g. First we go to the store

to buy a cake mix, then we will bake it, and after dinner we will eat it).

- ✓ Begins to comprehend simple logic puzzles (e.g. If I cut an apple in half, how many pieces would I have?)
- ✓ Independent play is longer (45 minutes or more).
- ✓ Plays cooperatively with peers for extended periods.
- ✓ Abstract thinking is becoming more advanced. For example, children this age can often comprehend the concept of "opposite". They can also complete simple analogies (e.g. Birds like to fly, fish like to \_\_\_\_\_).
- ✓ Uses some irregular past tense of verbs (e.g. ran instead of runned, left instead of leaved, fell instead of fall), but still over generalizes rules of grammar.
- ✓ Can play simple organized games, while remembering the rules (e.g. musical chairs).
- ✓ Enjoys pretend play with themes familiar to child (going to work, taking care of pets or babies, etc.).
- ✓ Can build relatively complex structures with blocks or LEGOs (houses, etc.).
- ✓ Fine motor skills are increasing. For example, by age 5 many children can operate difficult wind-up toys, or use a key.
- ✓ Can follow 3 simple directions in the appropriate sequence.
- ✓ Can answer questions about a short story.
- ✓ Can draw a person with 5 parts (e.g. head, hair, legs, arms, eyes).
- ✓ Can recognize letters and associate some letters with their sounds.
- ✓ Demonstrates understanding of one-to-one correspondence.
- ✓ Can complete puzzles with 8-12 pieces.
- ✓ Can copy a square.
- ✓ Can cut on curved lines.
- ✓ By age 5 can write own name.
- ✓ Can recognize numerals from 1 to 10.
- ✓ Can choose objects that have a similar characteristic, and express why they are similar.

### 5 to 6 Years

Kindergartners and 1st graders can use pull-down menus to launch programs themselves (some will even install them for you!). They can also use the computer for simulations, creativity and even for reference. With some help, they can go onto the Internet to research a topic of

interest, such as dogs, cats or that special pet lizard. This is a time when solid computer activities can play a valuable role in supporting and building school skills. By this age, children know where the keys are on the keyboard, and can hunt and peck their own names. But don't expect them to be able to type yet... formal typing skills will come much later.

I agree. My best stuff came from watching my own two kids.



- ✓ Understands the concepts of tomorrow and yesterday.
- ✓ Understands the concepts of morning and night.
- ✓ Knows his or her birthday month.
- ✓ Can tell time on the hour around age 6.
- ✓ Associates most letters with their sounds.
- ✓ Begins to recognize simple words.
- ✓ Knows both upper and lower case letters.
- ✓ Can match simple words with each other.
- ✓ Can answer "why" questions appropriately.
- ✓ Waits for turn while playing or while waiting for adult attention.
- ✓ Can follow the rules and directions of a classroom.
- ✓ Continues to engage in pretend play with themes familiar to child.
- ✓ Can adeptly use tools such as scissors, hammers, screwdrivers, etc. Can use scissors to cut out magazine pictures.
- ✓ Can use visual details to determine if two pictures are the same or different.
- ✓ Can copy a triangle.
- ✓ Can color pictures within the lines.
- ✓ Can write numerals from 1 to 10.
- ✓ Completes 10 to 15 piece puzzles.
- ✓ Can solve simple addition and subtraction exercises (If I had 4 apples and added 1 apple, how many would I have?).

# Section 3. Specific Tips

## Capturing the Magic of Interactive Media



If you want to design great interactive products for kids, it pays to have one. A kid, that is. Many case histories of excellent design begin when software programmers become parents, as was the case with Shelley Day. Mom Shelley wanted to find new ways for her son to play with his favorite homemade bedtime stories about a little car named “Putt Putt.” Long story made short— Putt-Putt became the “vehicle” that helped Humongous Entertainment become a \$60 million dollar company.

It's all about knowing and understanding kids. For both Mark Schlichting of the Living Books and Craig Hickman, who single handedly programmed the first version of Kid Pix, design success was born out of wanting to improve current products for their own children. Several years ago, Schlichting shared this story in CSR of how he became captivated by interactive technology.

“I'm a parent of three boys and I'd bring home what I thought was good learning software. Then my sons would play with it once, maybe twice and that was it. Around the same time, one of my older boys and his friends rented a Nintendo game. In the course of three hours they were up to the 52nd level of play. I thought to myself ‘Look how motivated these kids are to figure this out. There's an incredible amount of critical thinking going on, but in an environment with absolutely no content. Wouldn't it be great to use this natural draw to technology to deliver real learning through play and exploration?’” (A Conversation with Mark Schlichting, CSR March 1999)

Watching kids makes you wise. Schlichting's Living Books went on to become a standard-bearer of quality, loved by children, parents and teachers alike for their emphasis on good stories and entertaining exploration.

Kid Pix, the classic children's drawing program, was born out of frustration when programmer/photographer Craig Hickman saw his three-year-old son Ben struggling to use MacPaint. “I was surprised at how quickly he got the knack of using the mouse and how easily he was able to select tools. The problem was that he didn't have total control of the mouse and would occasionally (like every five minutes or so) pull down a menu and bring up a dialog box that he couldn't dismiss without being able to read. Everything was fine as long as I was in the room, but if I stepped out for a few minutes I would come back and find Ben kicking on the floor in frustration. This was not what I had in mind for his introduction to the computer.” (<http://pixelpoppin.com/kidpix/index.html> )

Thanks to Ben, Hickman went on to design Kid Pix, a rich, open-ended draw and paint program rating high in child control. Since Brøderbund's first publishing of the of the software in 1991, Kid Pix has been translated into dozens of languages and used by over ten million children around the globe. By CSR count, over 6,000 children's interactive media products have been developed in the last 15 years, and you can bet your bottom dollar that the best of these

were made by folks who know and love kids. By playing with and observing children— programmers, product managers, CEOs and even reviewers can learn some powerful lessons. To borrow from Bob Hughes' powerful book on interactive media, *Dust or Magic*, (Addison Wesley) some of the stuff is pure magic, while some is nothing but dust. Here are some of the ingredients of “magic” interactive products.

### A “Magic” Product...

**Is easy to set up.** Complex installation and registration routines on Windows computers have damaged the industry. Anything more than “Put the CD in the drive” should be outlawed. Hit Clips (by Tiger) exemplifies “ease of use”. Even the batteries are pre-installed. Just open the package, push the button, and it works.

**Lets kids “accidentally” succeed in the first 30 seconds.** Children, like grown-ups, want control! Early success in a program is like that great golf shot— it keeps you coming back for more. The program must provide the most direct path to what Hickman calls “the prime directive.” Take the typical racing game. You want to race cars, right? But some racing programs put roadblocks in between you and the racetrack, in the form of layers of customization menus. Let me race the car. Give me the preferences if I want them.

**Overdelivers and undersells.** Few products build customer loyalty faster than software. Parents and teachers can see the difference a well designed program makes for a child, and this builds an emotional bond to the product.

**Has a crisp, responsive interface that wants to please.** Each program takes a child into its own little world, with its own set of rules, and a distinct emotional climate. We tell designers to imagine their program as a “dinner date.” Some “dates” talk way too much, can't be interrupted, or don't remember things you'd already discussed. Author Bob Hughes offers another way to look at an interface, using a good dog as an analogy. Most computer interfaces are like “stuffed dogs”— static, they don't do anything. Other programs or toys are like hyperactive puppies—with so many writhing, flashing icons, they look like the Las Vegas strip. The “new & improved” Kid Pix 3 suffers from this fate, because an artist decided to over-stylize the icons at the expense of child control. A “good dog”

interface is alert, alive and “ready to help,” but it doesn’t detract from your attention. Developers forget that children are very tuned into subtle messages that they get from the program. Tiny delays in the action, non-intuitive icons, or sluggish reactions to a click can convert feelings of control into irritation.

**Is consistent.** Any teacher can tell you... when children know the rules, they settle down and are much less likely to misbehave. A good interface establishes the “rules” early on, and keeps them the same. Want to see a child throw a mouse? Make an icon that works only after the narration has stopped. Worse, make the same icon do different things, or put two “exit” icons on the same page. Hughes calls programs that change themselves around “Gestapo Interfaces” and compares the experience of trying to use a poorly designed web site (of which there are no shortage) with talking to a paranoid schizophrenic, where the rules can change at any time. “Do I click, or don’t I click? How do I get back to that screen I was playing a few minutes ago? Where’s the undo? ARGGGH!”

**Helps kids know where they are.** Most software uses some kind of “space” and that space needs to make sense to a child. One of the simplest approaches is to keep everything on one screen, like Space Invaders. Other tried and true techniques include an octopus-like storyboard, with different activities radiating out from a constant menu screen. Maps are another useful navigational tool. Programs like School Zone’s On Track series keep a constant navigation strip on the bottom of the screen. In this case, the strip is made of footprints, each one representing a page. The longer a child plays, the more footprints are filled in. Once they get to the edge of the screen, they’ve completed the book. Define the space, define the goal, and have a visible reminder of progress.

**Doesn’t underestimate or talk down to kids.** A wise Mississippi preschool teacher once said “young children can’t spell hypocrite, but they know what one is.” The same goes for software interfaces. Even very young children are smart when it comes to sniffing out the real play value of a product, especially programs that are dressed up in flash animations. Media pioneers such as the late Walt Disney understood that children can be the harshest critics. Kids respond to (and deserve) good art and music and original full-strength storylines. They don’t like being talked down to, patronized or underestimated. Software developers should be well versed in child development, so they understand what kids can do and when.

**Follows tried and true play patterns.** We call it “riding the horse in the direction it’s going.” Think about it. Why are RPG (role playing games) so successful? Simple— children love to pretend. RPG games are natural extensions of what kids are already doing. (Of course in our day it was cowboys and Indians, not warrior princesses and aliens!) Programmers and designers should spend some time at the playground. Things you notice there can end up as important elements in your products. Hickman writes “When Ben built something out of blocks, he enjoyed knocking his structure down almost as much as he did building it. Getting rid of the picture should be fun.” Hence Kid Pix’ exploding firecracker eraser, one of the greatest (and most controversial) menu tools ever created. Other good designs let children interact with products in ways that aren’t always intended, another hallmark of children’s play. Stretching a graphic or font way too big, piling up layers of stickers, creating chaos; are all part of sending the message to the child that they have control of this world.

**Offers social experiences.** Kids like games they can play together (ask Nintendo why they have four controllers). They also like games where they can print something out and share it with others, another important social opportunity. The Internet has connected computers, making the hard drive a virtual one. The Instant Messaging phenomenon, both on computers and handheld toys, is evidence of this “magic.” The ability to play basketball with another person across the country, or to drive a submarine in Castle Infinity (you steer, I’ll navigate), or capture the flag in Disney’s Atlantis, all create interesting contexts for socialization and group interactions.

We have a long way to go before all the bugs are worked out and these interactive social experiences are fully studied; but this trend is certainly the wave of the future and has “magic” written all over it.

## Enemies of Magic— (What Makes a Product “Dust”)

It’s not really accurate to classify all children’s interactive products into either “dust” or “magic.” Few are actually dust, most are fair, many are good. Only a dozen or so per year really stand out as “pure magic.”

### Those products that are “dust”:

- are hard to install
- talk too much
- offer too many symbols and abstractions that are not part of children’s past experiences
- sacrifice ease of use for “cool” design
- talk down to kids
- are boring, too hard, or too easy
- lack interactivity
- are inconsistent and confusing



# What all Software for Young Children Must Have

- ✓ Clear picture menus that do not require reading
- ✓ Simple “one layer” menus that provide direct access to activities
- ✓ Limited wait times to match short attention spans
- ✓ Quick, clear response to keystrokes
- ✓ Interruptible routines (e.g., opening sequences, animation, etc.)
- ✓ The ability to handle “machine-gun” keyboarding without buffer problems or crashes.
- ✓ Help that’s delivered via clear speech in the context of the problem (the program should not jump to a separate help sequence).
- ✓ Icons that are large, understandable to children (meaningful) and easy to select. Avoid using “phantom icons,” in other words, objects that ask to be clicked on but don’t do anything.
- ✓ Picture-driven printing and saving routines (not text-driven). Parents or teachers should have options for disabling the printing routines.
- ✓ Feedback/help that goes beyond simple reinforcement messages such as “nice job” or “try again.” The program may narrow the options (to increase the chance of success on a second try) or provide a hint to coach the child along.

## CHI-Kids: A Useful Listserve

<http://sigchi.org/sigchi/kids/chi-kids.html>

CHI-Kids@acm.org is for designers, producers, researchers, and practitioners of interactive media for children and adolescents. It is an online mailing list for discussing a range of issues related to the creation and implementation of kids’ media and technology. Children’s media professionals, researchers, and educators subscribe to CHI-Kids. CHI-Kids subscribers discuss a variety of topics, such as:

- How to define quality in children’s interactive and online software
  - Strategies for designing effective educational media
  - How girls and boys differ in their software preferences
  - Effects of violent interactive content on young people
  - Opinions about specific kids’ software titles
  - Problems and success stories related to implementing new technology with youngsters
  - Research methods and ethics
  - Research findings
  - New products and resources
- Job openings  
• Upcoming conferences and events

To Subscribe to CHI-Kids: Send a message to [listserv@acm.org](mailto:listserv@acm.org) and in the message area (not the subject area) type:  
subscribe chi-kids yourfirstname yourlastname  
For example: subscribe chi-kids Jane Smith

# A Dirty Little Secret: Many Kids’ Console Games are Too Hard

“I HATE this game!” screamed eight-year-old Sarah, as she tried to maneuver one of the Olsen twins down a snowboarding course. Two tries later, she threw down the controller and stormed to her room in tears.

Not exactly a pleasant experience. To make sure Sarah wasn’t negligent in her PSX skills, we gave the same game to our 18-year-old intern, who has logged in countless hours testing games. His assessment? “Tony Hawk, no problem. Mary Kate & Ashley? Too hard.”

Few would disagree with the importance of making a game challenging. It’s great to make a game where the player struggles—but the best products make sure players struggle with the right things, and not the wrong things. After reviewing several hundred console games for children, we’ve noticed that too many make kids struggle with tasks like entering their names, saving a game, selecting a language, or maneuvering a character. These tasks should be easy and intuitive. In the hopes of helping producers create better products, we’ve come up with this checklist for designing a successful children’s video game.

## Making Videogames for Kids— Design Tips 101

### DO

1. Guarantee success in the first 10 seconds of the experience.
2. Provide help or hints when a child struggles.
3. Use the controller buttons consistently.
4. Make it possible to get out of anything you can get into.
5. Create multiple paths to success.
6. Make sure the objective is clear.
7. Kid test, with a variety of children.

### DON'T

1. Start with a long series of instructions.
2. Assume the player is a seasoned video game player.
3. Use menus that require reading, especially if the program is designed for a preschool- or kindergarten-aged child.
4. Go light on the child testing.

So listen up, console programmers. Ease of use sells. Lose the jargon, get rid of confusing menus and be consistent with controller buttons. Don’t assume every child is a gamer, and be aware of children’s developmental levels when you create a game. Think of it as job security.

# Effects of Praise and Reinforcements on Engagement -- A Study of Two Interfaces on the Same Activity

[www.childrensoftware.com/dis/dis.menu.htm](http://www.childrensoftware.com/dis/dis.menu.htm)

There is an established body of research that has examined the interaction style between adults and children.

Some studies measured behavioral outcomes, such as various aspects of the educational effectiveness of the interaction. In the "wait-time" study, Mary Budd Rowe (1974) observed that the average time teachers waited between asking a question and taking further action to elicit a response is about one second. When a student responds to the question, teachers wait, on the average, less than one second before reacting to the response. Rowe called these two time periods-- the period between asking the question and acting further, and the period between the student's response and the teacher's reaction-- wait time. By asking teachers to increase their wait time to between three and five seconds, she observed a 300% increase in the length of students' explanations (Rowe, 1974).

Teacher/child interactions have been documented in intrinsic motivation literature (see Ames, 1990; Brophy, 1981; Lepper, 1985; Smilanski, 1968; Stipek, 1988 to name a few). Directly related to the study described in this dissertation is the literature that considers the quality and quantity of a child's engagement with a given task, as influenced by an adult/child interaction style. This relationship has been documented by Gerald Mahoney and James MacDonald (2003) with a population of young children with and/or at-risk for developmental problems. When children and parents or caregivers participated in two types of interactions (didactic and responsive), a positive relationship was identified between a responsive interaction style and children's social and linguistic development (Mahoney & MacDonald, 2003; Wolock, 1990; McWilliam et al., 2003).

This study examined these types of relationships in an interactive media context. A computer classification activity was created that was modified to simulate two contrasting teaching styles, similar to the Mahoney &

Are there observable differences in child behaviors in two versions of the same software sorting activity, one with a high level of instruction and reinforcement (high computer control), the other with relatively few instructions and reinforcements (high child control)?

**Designers and evaluators of interactive media products for children should pay careful attention to the degree to which the implementation of control mechanisms such as reinforcements can have substantial effects on children's interaction with the software.**

MacDonald technique. The first style, called "high computer control" (HICOMP) attempted to simulate a teaching style where the teacher carefully introduced each problem, and provided frequent praise and encouragement throughout the experience. As a result, the child had less control over the flow of events, making the experience less responsive. The second style, called "high child control" (HICHILD) presented the identical sorting experience with the instructions, praise and encouragement turned off. As a result, a child experienced more control over the events, resulting in a more responsive overall experience.

Control, which was varied by changing the quantity of instructions and reinforcements, served as the independent variable in the study. The dependent variable, the engagement of the child, was measured by counting observable child behaviors that were recorded on videotape. These included 1) the number of tasks completed, 2) the number of clicks, or attempts to influence the instruction flow, and 3) the length of time the child chose to spend with each condition.

The study population was 38 pre-school-aged children. In addition to discrete continuous measures, anecdotes during data collection provided additional insights. What children said and how they behaved throughout the experience was also documented, specifically children's competency with the input device (a mouse), the ways in which they responded to the software interface, and any other observations that could possibly be related to their engagement with the activity.

## The Results in Brief

The outcome variables were the number of tasks attempted, tasks correct, time with the activity, mouse clicks and a child rating of the experience. In addition, anecdotal observations documented child reactions to both settings.

**Children in the high child control treatment were more active, completing more tasks (mean = 64 vs. 20;  $p < .05$ ), clicking the mouse more times (mean = 129 vs. 73;  $p < .05$ ), and getting more tasks correct (mean = 41 vs. 16;  $p < .05$ ). Children rated both experiences highly, and spent about the same amount of time with each condition.**

In the high computer control setting, there were more clicks per task (mean = 4.07 vs. 2.09;  $p < .05$ ), and children had a higher accuracy level (mean = 85% vs. 68% respectively). In addition, ANOVA procedures suggested that younger children choose to stay with the HICOMP experience longer than the older group of children.

This study helps connect the established principles of human/child interaction to computer/child interaction, including the role of external reinforcements and the level of responsiveness of the interaction. The results of this study suggest that designers and evaluators of interactive media products for children should pay careful attention to the degree to which the implementation of control mechanisms such as reinforcements can have substantial effects on children's interaction with the software.

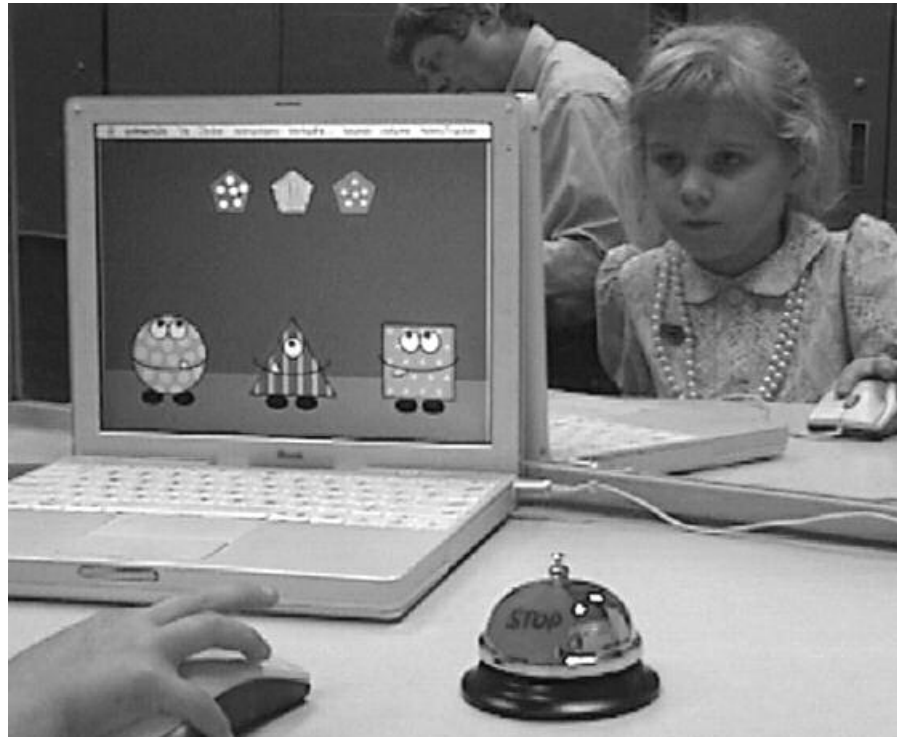
## A Closer Look

The measures revealed some interesting, statistically significant ( $p < .05$ ) relationships. Children in the high child control setting performed more mouse clicks (129 vs. 73) and had lower accuracy rate for problems (68% vs. 85%), in about the same amount of time.

However, the most striking finding was that children attempted over three times more problems (64 vs. 20) and more than twice as many correct answers (41 vs. 16) in the high child control condition. While no significant differences were found by gender or session administration, the age of the children did matter in terms of the amount of time spent with the task. The population's large developmental span (40 to 60 months) led to a high degree of variability in children's performance across the tasks. In addition to the expected differences due to cognitive abilities, some children seemed determined to test the outer limits of the activity. Considering that outliers were not removed from the data set and the population of young children was heterogeneous, especially with regard to developmental level, the resulting standard deviations for time, clicks and tasks were large.

## Major Findings

**1. Children clicked more in the HICHILD setting, but had fewer wasted clicks than in the HICOMP setting.** For the purposes of this study, a click is defined as the two part motion (and up and down stroke) when children choose to interact with the software interface. The click was easily counted due to the distinctive sound associated with stroke, as well as the visual clues provided by screen events. Mouse clicks were a useful behavioral indicator of a child's motives, interests, ability level and activity level. This meaning of the click depended on the situation. The click could mean something either intended, or non-intended. Intended clicks might have a meaning such as "OK, I'm ready, lets get started", "I'd like to choose that cookie", "Is this cookie the right one?" or an expression of affect such as "hey, hurry along" or "I'm feeling very frustrated!" In the high child control setting, children clicked more (mean = 129.08 vs. 73.68



respectively;  $p < .05$ ) over the same amount of time as the high computer control setting. This outcome has more meaning when interpreted in the context of the number of problems completed in each setting. In the HICHILD setting, children attempted more than three times (320%) the number of tasks (63.8 vs. 20.4;  $p < .05$ ), resulting in a click per task ratio nearly two times (194%) that of the HICOMP setting (4.07 vs. 2.095;  $p < .05$ ).

To conclude, when responsivity was increased, children were much more active, clicking more frequently; and more of those clicks were related in some way to an intended outcome (from the perspective of the software designer). In the HICOMP treatment, the added narration and reinforcement statements seemed to create a barrier to child's activity and problem solving effort.

**2. Children chose to spend about the same amount of time in each trial.** But there were interesting interactions for time and age. One initial hypothesis of this study was that the lower the level of responsivity, the less time a child would voluntarily stay with treatment. Contrary to this prediction, children in

the HICOMP setting actually spent a bit more time than when in the HICHILD condition (mean = 480 seconds vs. 540 seconds, respectively) although this relationship was not significant ( $p > .05$ ). The novelty of the sorting experience kept the children in both situations, and children's willingness to comply with the software's instructions might explain the slightly greater time spent in the HICOMP situation. Because both treatments were conducted out of a child's regular play setting in a parent lounge down the hall, it is possible that the novelty effect was magnified. This could be further explored with an additional study that would examine child reactions to each treatment in the context of a free choice period, over the course of a longer period of time, along with a hidden counter in the software to measure the frequency of use. In this study, these questions were not formally addressed.

The ANOVA revealed some notable findings when the entire group of children was divided by younger and older age groups. The 14 younger children, aged < 50 months on average chose to stay with the experience longer than the 22 older children ( $p < .05$ ) regard-

less of the experimental condition. An explanation for this may be the challenge level, which started with three objects to sort, based on one attribute, and increased to five objects and three attributes. Because most of the problems were geared toward the middle of the age group (46 to 52 months), the older, more competent children more quickly exhausted the novelty and challenge available in the experience than the younger group, resulting in a loss of interest, and less time on task. For designers, this helps illustrate the importance of having a fluid challenge level that either automatically adapts to the child's ability level, or that lets the child have some control over the challenge setting.

**3. Children attempted more problems and experimented more in the HICHILD setting.** Another statistically significant relationship was the number of problems that were attempted between the two treatments. In the HICHILD condition, the children were 317% busier, attempting 63 problems in approximately the same amount of time spent in the HICOMP condition with only 20 problems solved ( $p < .05$ ).

When children experienced a more structured and controlled interface with a high level of narration and direction, they showed a decrease in activity, as measured by number of problems attempted. Anecdotal observations supported this observation, with more fidgeting, yawning, and placing head on the table during the HICHILD situation.

Another observation relevant to this topic was that the HICOMP treatment work was more accurate, with a higher percentage of correct answers (84.95% vs. 67.97% respectively;  $p < .05$ ). When there was increased activity, there was a decrease in accuracy. When the sum of correct answers, however, was compared between the two conditions, during the HICHILD condition children ended up with 393% more correct answers -- 41.0 vs. 16.1 ( $p < .05$ ).

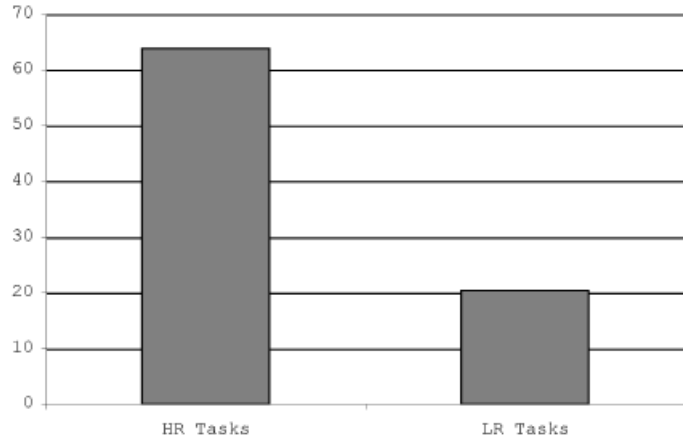
Interpreting the significance of this finding is dependent upon the theoretical framework and associated instructional objectives of the software designer. If the end goal is for the learner to solve a higher number of correct answers

and increase the amount of experimentation, the HICHILD setting is the preferable design. If higher accuracy regardless of the number of problems is the only goal, the HICOMP setting is the preferable option.

**4. The older group of children chose to spend less time in the HICHILD setting than the younger group of children.** When the population was grouped into two parts by age (over 50 months and under 50 months), there was a significant and interesting difference in the amount of time the two groups choose to stay with the activity. Regardless of the experimental condition, the younger group stayed longer than the older children ( $p < .05$ ) although the HICHILD setting held them longer. (610 vs. 442 seconds, whereas the HICOMP setting was 573 vs. 567 seconds).

It may be that the increased responsiveness of the HICHILD setting allowed the older, more competent children to use up the challenge sooner, resulting in a shorter session time. The greater number of attempted problems by the older children (80.57 vs. 52.10) supports this observation, particularly when compared with the older children in the HICOMP setting, where the difference narrowed to 20.6 vs. 20.2. The decreased responsiveness seemed to be associated with the child's ability to quickly reach their challenge level, which had an affect on the amount of time they chose to stay with the activity.

**5. Children rated both experiences highly, but anecdotal observations seemed to indicate that children generally preferred the HICHILD treatment over the HICOMP treatment.** A formal measure of the child's feelings about each treatment was attempted using a Likert-type scale. There were no significant differences between the two groups (4.65 for HICHILD, 4.58 for HICOMP;  $p > .05$ ). When children were asked "how did you like it?" immediately after a treat-



ment, they would say either nothing or that they liked it, by touching one of the smiles faces. It was hypothesized that children would rate the HICOMP experience lower than the HICHILD setting. This was not supported by the survey ratings.

Additional information was gathered less systematically, by observing children's reactions when their turn came up to play the second trial. In general, they would respond enthusiastically to the idea of coming back to the room to play the game some more, regardless of the first condition they experienced; high or high computer control. Nine of the children, generally older, were able to verbally compare the HICHILD and HICOMP treatments after the second session. From these videotaped conversations, it was possible to determine that these children had more positive things to say about the HICHILD experience. In order to more accurately understand children's reactions to each treatment, additional exposures to both the HICHILD and HICOMP treatments would be necessary, over a longer period of time. It is likely that children would have a more discriminating attitude toward between the two treatments after the novelty of the experience is reduced.

# Eight Lessons for Structured (Didactic) Activities

The following elements of the Cookie Critters experience appeared to make a difference in the quality of the child's time with the Cookie Critter's activity. These non-systematic observations were taken as notes during the each administration session and while coding the tapes.

**1. Include a brief, ten second "launching experience."** The importance of providing a launching event, or an "anticipatory set" (Hunter, 1982) that could get a child's attention (Gagné, 1977) and then provide a clear path for the child to take the next transaction in the interaction played a key role in a child's initial reaction to the Cookie Critters activity. In both the HICHILD or HICOMP treatment, a short, one sentence phrase such as "click a cookie" that is spoken as the clickable cookie is highlighted on the screen, advertised what was needed to do in order to get started. When the launching instructions were toggled off, or when they were set on the maximum setting, the younger children seemed more likely to become lost or distracted.

**2. Insure quick success for every child, regardless of developmental level.** Approximately five of the 41 children were resistant to participation. This may have been due to some past unsuccessful experience with a computer activity, but it is important to note that approximately 12% of this particular sample seemed to feel strongly that computer activities were not something for them. When the first few screens and the introductory sequence were short, clear and easily bypassed, children seemed more likely to experience some degree of "accidental success." There was one flaw that was identified in the Cookie Critter's activity that affected ease of use for several children. The first screen starts with an inch-wide round target with the printed word "start" on it (Figure 10). In order to unlock the activity screen, children are required to hit this target, which implies that they will know that the button means start. While this is logical to an adult who can read, a preschool child can't, so there is no indication what to do. One way around this bottleneck would be to make the screen so that any click, regardless of where the

cursor is, advances the program to the next screen.

**3. Incorporate dynamic, or "living" features that are driven by, or respond directly to the child's actions.** In the starting screen in Figure 10, for example, children would be more likely to become engaged early on if the eyes in word "Cookie" followed the cursor around the screen. Dynamic animation properties such as these, that follow the initiation of the child, seem to be effective for increasing initial engagement.

**4. Use humor carefully and intermittently.** Children seemed to respond well to events such as when the critter burped after eating a correct match, and "bonk" sounds when cookie did not match. These small events worked very well to support children's engagement.

**5. Opt for context sensitive "roadside assistance" in place of lengthy segments of spoken instructions.** Ideally, interactive media products designed for young children could be able to sense outlier behaviors, such as series of errors, and respond appropriately. This assistance cannot disrupt the current activity; for example, by launching a new path with a help sequence. It needs to happen out of the way, while respecting the child's current problem solving space.

**6. Put children in the role of being in control.** In this case, it was being able to be in charge of feeding the cookies, determining which cookie was able to eat, and which wasn't.

**7. Capitalize on a children's initial motivation.** Each child started both HICHILD and HICOMP experiences with some level of motivation. It is up to the designer to determine how this motivation will be spent. This study illustrated that this motivation can be either used for more accurate responses and less activity, or more activity with more correct answers and more mistakes, depending on quantity of the reinforcements and instructions. In the case of an activity like Cookie Critters, it is a question of the instructional design priorities.

**8. Provide a meaningful context, from the perspective of the child, not an adult.** The

first administration session used a version of Cookie Critters with a visual progress tracking feature turned on. This made it possible for children to see how many problems they had solved, and how many more they had to do before the next challenge level. This technique has been used in commercial software products, including Stickybear Math (Optimum Resources) and such as School Zone's "On Track" series of software.

## Classification of Mouse Clicks Listed by Frequency During the Cookie Critters Activity

This is an attempt to classify the types of mouse use observed in this study.

**1. Double Stroke, Intentional Clicks.** This click consisted of one complete down and up stroke while on the intended target. For example, the child sees a cookie, moves the cursor to it, and clicks. This type of click was more common in the older group of children (>50 months) who were more likely to have prior mouse experience. This type of click was common in both HICOMP and HICHILD settings.

**2. Single Stroke, Intentional Clicks.** Approximately 1 in 5 children used "drag and drop" or "hold and go" (Strommen) single stroke clicks in both the HICHILD and HICOMP settings, even though the activity used a "sticky mouse" making this technique unnecessary. A child using this type of click would first position the cursor over the target cookie, and then make one downstroke, holding down the mouse button, and not letting it come back up until it was over the target critter. This type of click requires the coordination of both fine motor and gross motor movements simultaneously. It was interesting that some children switched to this strategy in the HICOMP setting, from intentional clicks, after they learned that they could not speed the events along. Perhaps this was out of frustration.

**3. "Hurry Up!" Unintentional Clicks.** This click resulted when a child attempts to influence the temporal sequence of events on the screen by clicking the mouse. Commercial early childhood software activities that allow children to "click through" introductions or screen events may reinforce this behavior. This clicking behavior was observed only in the HICOMP setting.

**4. "Rapid Fire" or "Machine Gun" Unintentional Clicks.** This technique refers to when child sends a continuous stream of clicks, sometimes in a short burst and other times for longer sequences. The child's thinking seems to be along the line of "I'll just keep clicking until the computer hears me." It was also a way to keep busy, perhaps creating a simulated feeling of control in the HICOMP setting. This was rare in HICHILD settings, much more common in the HICOMP setting when children did not have as much control.

# CIMEI: Children's Interactive Media Evaluation Instrument

Title \_\_\_\_\_  
 Publisher \_\_\_\_\_  
 Copyright Date \_\_\_\_\_  
 Publisher's Phone/URL \_\_\_\_\_

Price \_\_\_\_\_  
 Platform \_\_\_\_\_  
 Ages \_\_\_\_\_  
 Teaches \_\_\_\_\_

**Instructions:** Spend a few hours testing all aspects of the interactive product, preferably with a child, making note of key strengths and weaknesses. Then use this instrument, and calculate your rating.

**I. Ease of Use (Can my child use it with minimal help?)**

- Always SE Never NA
1. \_\_\_\_\_ Skills needed to operate the program are in range of the child
  2. \_\_\_\_\_ Children can use the program independently after the first use
  3. \_\_\_\_\_ Accessing key menus is straightforward
  4. \_\_\_\_\_ Reading ability is not prerequisite to using the program
  5. \_\_\_\_\_ Graphics make sense to the intended user
  6. \_\_\_\_\_ Printing routines are simple
  7. \_\_\_\_\_ It is easy to get in or out of any activity at any point
  8. \_\_\_\_\_ Getting to the first menu is quick and easy
  9. \_\_\_\_\_ Controls are responsive to the touch
  10. \_\_\_\_\_ Written materials are helpful
  11. \_\_\_\_\_ Instructions can be reviewed on the screen, if necessary
  12. \_\_\_\_\_ Children know if they make a mistake
  13. \_\_\_\_\_ Icons are large and easy to select with a moving cursor
  14. \_\_\_\_\_ Installation procedure is straightforward and easy to do
- \_\_\_\_\_ TOTAL EASE OF USE

**II. Childproof (Is it designed with child-reality in mind?)**

- Always SE Never NA
1. \_\_\_\_\_ Survives the "pound on the keyboard" test
  2. \_\_\_\_\_ Offers quick, clear, obvious response to a child's action
  3. \_\_\_\_\_ The child has control over the rate of display
  4. \_\_\_\_\_ The child has control over exiting at any time
  5. \_\_\_\_\_ The child has control over the order of the display
  6. \_\_\_\_\_ Title screen sequence is brief or can be bypassed
  7. \_\_\_\_\_ When a child holds a key down, only one input is sent to the computer
  8. \_\_\_\_\_ Files not intended for children are safe
  9. \_\_\_\_\_ Children know when they've made a mistake
  10. \_\_\_\_\_ This program would operate smoothly in a home or classroom setting
- \_\_\_\_\_ TOTAL CHILDPROOF

**III. Educational (What can my child learn from this program?)**

- Always SE Never NA
1. \_\_\_\_\_ Offers a good presentation of one or more content areas
  2. \_\_\_\_\_ Graphics do not detract from the program's educational intentions
  3. \_\_\_\_\_ Feedback employs meaningful graphic and sound capabilities
  4. \_\_\_\_\_ Speech is used
  5. \_\_\_\_\_ The presentation is novel with each use
  6. \_\_\_\_\_ Good challenge range (this program will grow with the child)
  7. \_\_\_\_\_ Feedback reinforces content (embedded reinforcements are used)
  8. \_\_\_\_\_ Program elements match direct experiences
  9. \_\_\_\_\_ Content is free from gender bias
  10. \_\_\_\_\_ Content is free from ethnic bias
  11. \_\_\_\_\_ A child's ideas can be incorporated into the program
  12. \_\_\_\_\_ The program comes with strategies to extend the learning
  13. \_\_\_\_\_ There is a sufficient amount of content
- \_\_\_\_\_ TOTAL EDUCATIONAL VALUE

**IV. Entertaining (Is this program fun to use?)**

- Always SE Never NA
1. \_\_\_\_\_ The program is enjoyable to use
  2. \_\_\_\_\_ Graphics are meaningful and enjoyed by children
  3. \_\_\_\_\_ This program is appealing to a wide audience
  4. \_\_\_\_\_ Children return to this program time after time
  5. \_\_\_\_\_ Random generation techniques are employed in the design
  6. \_\_\_\_\_ Speech and sounds are meaningful to children
  7. \_\_\_\_\_ Challenge is fluid, or a child can select own level
  8. \_\_\_\_\_ The program is responsive to a child's actions
  9. \_\_\_\_\_ The theme of the program is meaningful to children
- \_\_\_\_\_ TOTAL ENTERTAINMENT VALUE

**V. Design Features (How smart is this program?)**

- Always SE Never NA
1. \_\_\_\_\_ The program has speech capacity
  2. \_\_\_\_\_ Has printing capacity
  3. \_\_\_\_\_ Keeps records of child's work
  4. \_\_\_\_\_ "Branches" automatically: challenge level is fluid
  5. \_\_\_\_\_ A child's ideas can be incorporated into the program
  6. \_\_\_\_\_ Sound can be toggled or adjusted
  7. \_\_\_\_\_ Feedback is customized in some way to the individual child
  8. \_\_\_\_\_ Program keeps a history of the child's use over a period of time
  9. \_\_\_\_\_ Teacher/parent options are easy to find and use
- \_\_\_\_\_ TOTAL DESIGN FEATURES

**VI. Value (How much does it cost vs. what it does? Is it worth it?)**

Considering the factors rated above, and the average retail price of software, rate this program's relative value.

Poor.....Good  
 1 2 3 4 5 6 7 8 9 10

\_\_\_\_\_ TOTAL VALUE

*Adding Up the Scores*

**Step 1:** Count the totals under each column.

**Step 2:** Assign point values, as follows:  
 Always = 1 point each  
 Some Extent = 1/2 point each  
 Never = 0 points each  
 NA = Not counted

*You can use this formula*  

$$\frac{(X + (Y/2))}{n - Z} \times 100 = S$$

*Where*  
 X = Total of checks in the "always" column  
 Y = Total of checks in the "some extent" column  
 Z = Total of checks in the "NA" column  
 n = Number of items in a category (such as Childproof)  
 S = Score for a component of the program (as a percent)

**Step 3:** Get each Component Score, or S, add them up, and use the formula below to calculate the overall star rating.

\_\_\_\_\_ Ease of Use  
 \_\_\_\_\_ Childproof  
 \_\_\_\_\_ Educational Value  
 \_\_\_\_\_ Entertainment Value  
 \_\_\_\_\_ Design Features  
 \_\_\_\_\_ Overall Value

\_\_\_\_\_ ÷ 6 = (\_\_\_\_\_ x 5) ÷ 100 = \_\_\_\_\_ stars

**Comments:**

\_\_\_\_\_

**Please note.** Permission is granted to use this instrument for educational use, e.g., teacher training. All other uses or adaptations must be done with permission only. Please contact CSR at 908-284-0404. © 2002 *Children's Technology Review*